B. Tech. Trimester XII

CAPSTONE PROJECT FINAL REPORT

On

Kinematic Modelling and motion mapping for control of Robotic arms in a ROV

Submitted by

Anish Agarwal- 1032170830

Ajinkya Chaudhary-1032170855

Anand Ganthale-1032171683

Project Guide: Prof. Mangesh Saraf

Year: 2020-2021

School of Mechanical Engineering

MIT World Peace University, Pune

ACKNOWLEDGEMENTS

Successful realization of this project and preparation of Project Stage report entails to many energetic and enthusiastic minds. We would like to take this opportunity to thank all of them, without whom, this stage would not have been successfully completed.

First, we would like to thank our sponsor, AKER SOLUTIONS, because of whom we received this opportunity to accomplish our B.Tech. project and hence partially contributing in their endeavour of building Subsea Laboratory in the campus of MIT, Pune. We would like to give special thanks to our industry team viz., Mr. Parag Paranjape, and Mr. Satyajit Lonkar for their consistent support and help in making this project successful.

Secondly, we would like to express sincere gratitude towards our beloved guide, **Prof. Mangesh Saraf**, who encouraged and enlightened us in many walks towards our grail in finalizing the design.

We would also like to thank our Head of Department **Prof. Dr. S. T. Chavan** who stood behind us in every mode of our project to make it successful.

Last but not the least, the consistent support by our staff members, our colleagues, and our parents for cheering and motivating us, share an equal half of this success.

CERTIFICATE

SCHOOL OF MECHANICAL ENGINEERING

This is to certify that ***Mr. Anish Agarwal (1032170830) has*** successfully completed the Project entitled **"Kinematic Modelling and motion mapping for control of Robotic arms in a ROV"** under my supervision, in the partial fulfillment of Bachelor of Technology - Mechanical Engineering of Dr. Vishwanath Karad MIT World Peace University, Pune.

Date:

Place:

Prof. Mangesh Saraf Project Guide	Examiner
Prof. Dr. S. T. Chavan Head, School of Mechanical Engineering	Prof. Dr. P. D. Khandekar Dean, FoET

CERTIFICATE

SCHOOL OF MECHANICAL ENGINEERING

This is to certify that ***Mr.*** **Ajinkya Chaudhary(1032170855)**

has successfully completed the Project entitled **"Kinematic Modelling and motion mapping for control of Robotic arms in a ROV"** under my supervision, in the partial fulfillment of Bachelor of Technology - Mechanical Engineering of Dr. Vishwanath Karad MIT World Peace University, Pune.

Date:

Place:

Prof. Mangesh Saraf
Project Guide

Examiner

Prof. Dr. S. T. Chavan
Head, School of Mechanical Engineering

Prof. Dr. P. D. Khandekar
Dean, FoET

CERTIFICATE

SCHOOL OF MECHANICAL ENGINEERING

This is to certify that ***Anand Ganthale-(1032171683)*** has successfully completed the Project entitled **"Kinematic Modelling and motion mapping for control of Robotic arms in a ROV"** under my supervision, in the partial fulfillment of Bachelor of Technology - Mechanical Engineering of Dr. Vishwanath Karad MIT World Peace University, Pune.

Date:

Place:

Prof. Mangesh Saraf
Project Guide

Examiner

Prof. Dr. S. T. Chavan
Head, School of Mechanical Engineering

Prof. Dr. P. D. Khandekar
Dean, FoET

Certificate of consultancy

This is to certified that the following students have successfully completed their B. Tech project title **Kinematic Modelling and motion mapping for control of Robotic arms in a ROV** as a partial completion to the parent project title **design and development of ROV for Subsea laboratory** during the academic year 2020- 21.The subsea Laboratory is being developed by MIT Pune in collaboration with Aker solutions Pune, where the funding of parent project is a corporate social responsibility initiative.

We Aker solutions are glad for the satisfactory correspondence with the project group through all stages of project.

Project members: -

1. Anish Agarwal- 1032170830
2. Ajinkya Chaudhary-1032170855
3. Anand Ganthale-1032171683

Project Guide	Industrial Guide
Prof. Mangesh Saraf	Mr.Prag Pranjape
SoME MIT-WPU	Project Manager CSR
	Aker Solutions

ABSTRACT

A ROV is a remotely operated underwater vehicle which is basically a tethered underwater mobile device. ROVs are an important tool in understanding underwater environments and to repair the complex deep water production systems in oil and gas industry. Design of navigation system for our ROV includes selection of appropriate mechanism and suitably modifying it as per requirements. The navigation system of our ROV is equipped with ballast tanks for balancing and thrusters for horizontal motion, vertical motion and rotation about Z-axis. This involves use of Inertial Measurement Unit as a feedback element. This system is optimized by trade-off between power consumption, complexity in control system and cost. Moreover, the navigation System of the ROV includes cameras and lights for real time visual feedback to the operator, while joystick in the hands of operator provides input control signals to microprocessor fitted on the ROV. For performing various operations, the ROV is equipped with two robotic arms also called as manipulators. One manipulator will have the function of grabbing, or clamping, while the other manipulator will serve the tooling functions of turning, pushing, picking, placing, etc. The ROV is designed to bear the load of both the arms and stabilize itself during operation of the arm. Robotic systems are known for their precision and accuracy. To achieve a desired set of operating parameters the system must have a good control system. With increasing DOF the complexity of the system increases and hence a DAQ system is required to check various parameters of the process. The forward kinematic model is based on Denavit-Hartenberg parameters. To obtain a given set of position and orientation of end effector, inverse kinematics gives the required set of joint angles. Potentiometers mounted on arm joint are used as the prime feedback elements for extracting the joint angles. All six joints of the arm are revolute joints. The values of potentiometer obtained are compared to the desired input values. Proportional direction control valve is the main controlling element of the hydraulic circuit. The forward and inverse kinematics is validated and implemented in MATLAB robotics toolbox (developed by Peter Croke). A GUI is also used to ease the process of retrieving and feeding the information in the system

Table of Contents

List Of Figures

List Of Tables

NOMENCLATURE

ρ : Density (kg/m^3)

A : Projected Area of Body (m^2)

ROV : Remotely Operated (underwater) Vehicle

g : Acceleration due to gravity (m/s^2)

IMU : Inertial Measurement Unit

C.O.G. : Centre of Gravity

C.B. : Centre of Buoyancy

F_b : Buoyant force

IP : Ingress Protection

V : Volume (m^3)

C_d : Drag Coefficient

v : velocity (m/s)

m : mass (kg)

T : Time (sec)

PVC : Polyvinyl chloride

S_{yt} : Tensile yield strength

d_c : Core diameter of bolt

d : Nominal diameter of bolt

p : Pitch

z : Number of threads

σ_c : Compressive stress

DOF : Degree of Freedom

GPH : Gallon per Hour

1.Introduction

Remotely Operated Underwater vehicle is an unmanned tethered robot that allows the vehicle's operator to remain in a comfortable environment while ROV works in hazardous environment at the sea floor. The total ROV system is comprised of the vehicle, which is connected to the control van and the operators on the surface by a tether or umbilical - a group of cables that carry electrical power, video and data signals back and forth between the operator and the vehicle - a handling system to control the cable dynamics, a launch system and associated power supplies. Most ROVs are equipped with at least a video camera and lights. Additional equipment is commonly added to expand the vehicle's capabilities. These may include sonars, magnetometers, a still camera, a manipulator or cutting arm, water samplers, and instruments that measure water clarity, light penetration and temperature. ROVs can vary in size from small vehicles with TVs for simple observation up to complex work systems, which can have several dexterous manipulators, display screens, video cameras, tools and other equipment. Today, advanced technology is allowing many ROVs to shed their cable, and thus become free to roam the ocean without such physical constraints. These emerging systems, which are battery operated, are called autonomous underwater vehicles (AUVs) and are used for ocean search and oceanographic research.

Figure 1:Actual ROV in Sub Sea

1.1 Subsea technology and subsea industry

Subsea is a term used to describe fully submerged ocean equipment, operations or applications, especially when some distance offshore, in deep ocean waters, or on the seabed. The term is frequently used in connection with oceanography, marine or ocean engineering, ocean exploration, remotely operated vehicle (ROVs) autonomous underwater vehicles (AUVs), submarine communications or power cables, seafloor mineral mining, oil and gas, and offshore wind power.

1.2 Scope

A work class ROV is to be designed and fabricated for a subsea lab being developed by Aker Subsea Power gas Pvt. Ltd. At Maharashtra Institute of Technology, Pune. The project is being sponsored by Aker under Corporate Social Responsibility (CSR). The project is an interdisciplinary project comprising of 11 members (6 of mechanical and 5 of Electronics). The ROV will be operated in a water tank of given specifications. The ROV is expected to be able to manoeuvre through the underwater assemblies and operate upon the specific assembly. The working conditions and some characteristics of the ROV to be designed and fabricated are as follows.

· ROV is to be designed to operate in fresh water and still environment.

· ROV is to be equipped with camera and lights for continuous video monitoring and underwater visibility while performing operations.

· The depth of navigation is up to 3 meters.

· Expected life of the ROV is 15 years.

· ROV is to be equipped with two arms for working and grabbing.

· The Degrees of Freedom of the arms is 6.

· The ROV will have controlled stabilization about X, Y, Z axes.

2.Problem Statement

Kinematic Modelling and Motion Mapping of Robotic Arms

2.1 Objectives

- Forward kinematics of 4dof and 6dof arm
- Inverse kinematics of 4dof and 6dof arm

- Mapping and control of PDCV
- Weight optimization of robotic arms

3.Working Parameters

3.1Working environment

The dimensions of the tank are 10m*10m*3m. The tank will be holding a small-scale oil extraction assembly on which the ROV is supposed to perform prescribed tasks. The ROV will be lowered into the tank using a skid from one end of the tank.

.

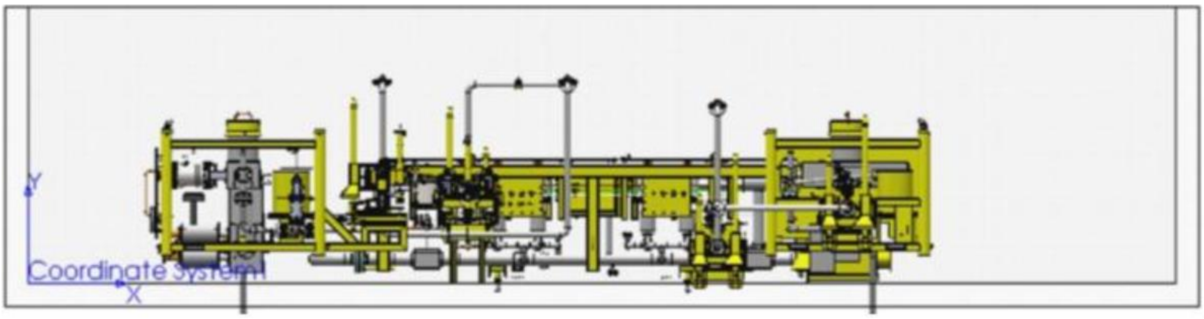


Figure 2: Working environment

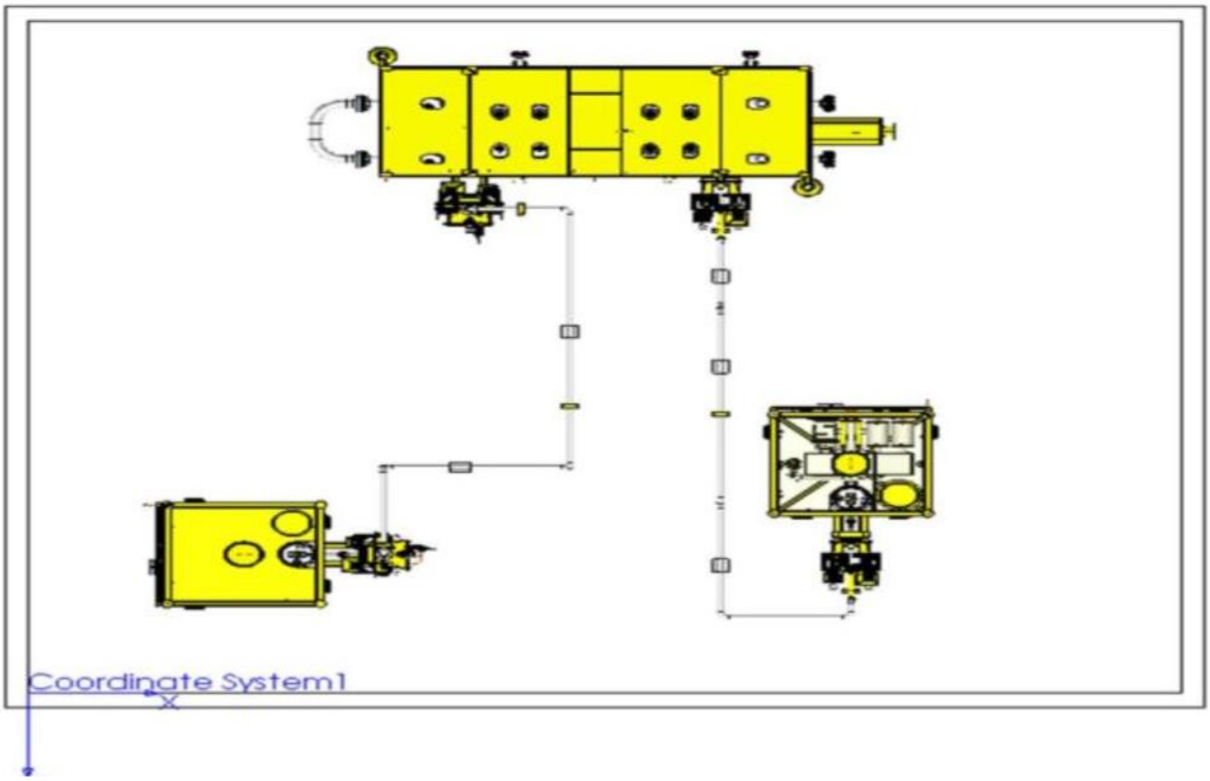


Figure 3:Working Environment of the vehicle

The ROV will be always connected to an overhead crane as a safety measure. The crane will ensure that the ROV will not sink in case of power outage and also to recover the ROV in case the ROV fails.

The ROV will be controlled from an electronic station placed outside the tank.

3.2 ROV specifications

Considering the space available for the ROV to maneuverer through the manifold assembly, the ROV is to be built within a specified envelope.

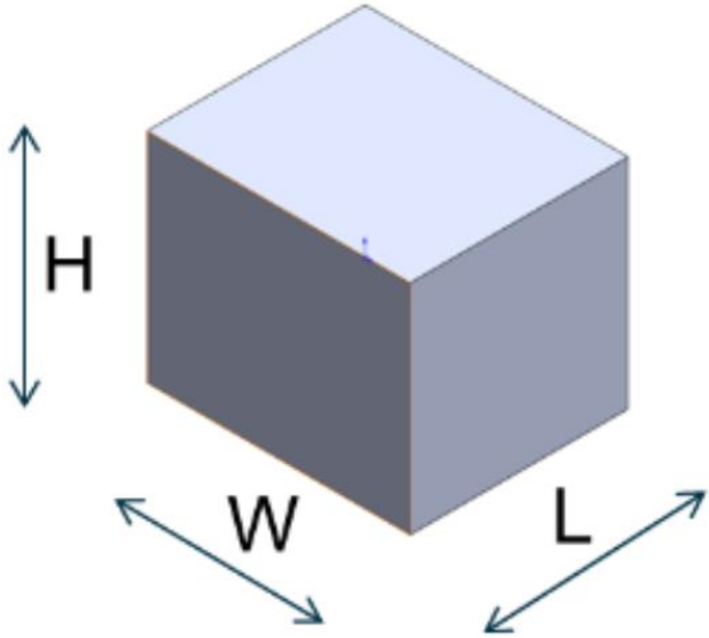


Figure 4:Rov Envelope

L=1170, W=970 and H=970

The ROV must fit inside the envelope while the arms are retracted. While operating on the manifold assembly the arms will extend outside the envelope.

3.3 Manifold Assembly (Conventions and naming)

In petroleum and natural gas extraction, a Christmas tree, or "tree", is an assembly of valves, spools, and fittings used for an oil well, gas well, water injection well, water disposal well, gas injection well, condensate well and other types of wells. The oil is extracted using the Xmas tree. A manifold is placed in the centre of a network of Christmas tree. Each Christmas tree sends the oil collected to the manifold, from where, it is sent to the ship or stations above sea level.

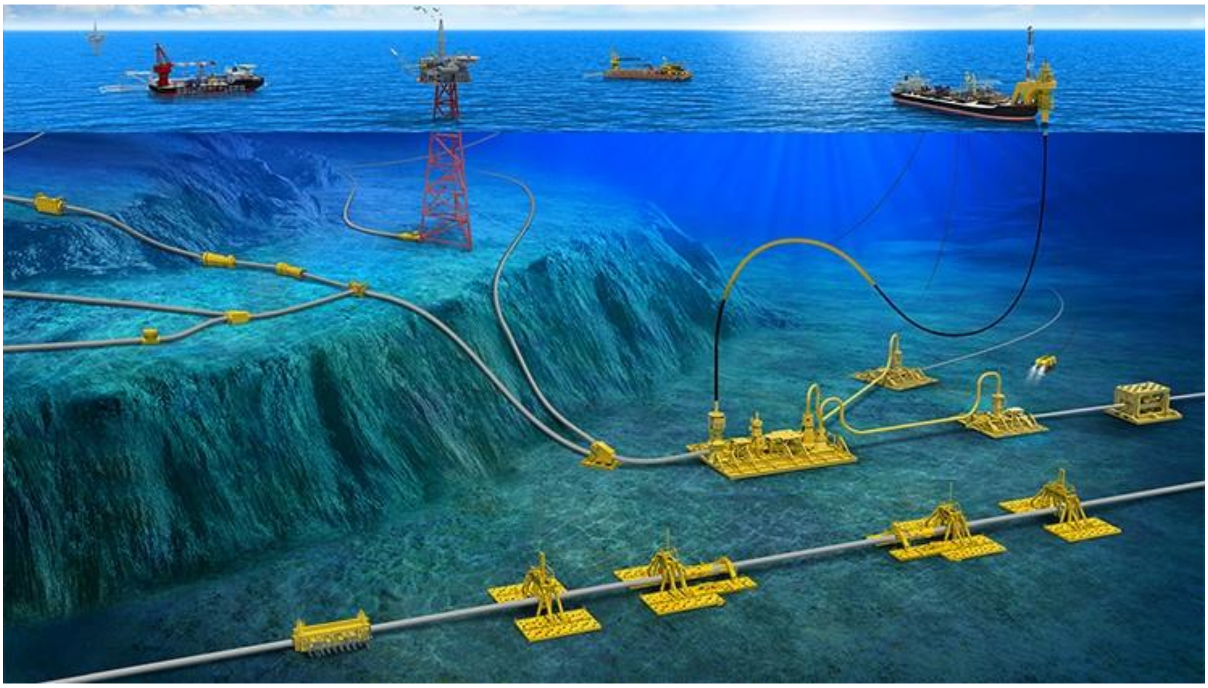


Figure 5: Industrial Manifold and Xmas tree system for oil extraction

The water tank will contain an assembly of two Christmas trees and a manifold. The oil extracted from the Xmas tree is sent to manifold through VSCs. There are two VSCs each on Xmas tree and manifold for sending and receiving, respectively. It is the VSCs on which the operations are to be performed.

The assembly to be setup at the lab is as shown in figure 4.

The ROV will be operating on VSC on Xmas tree 1 and VSC on manifold.

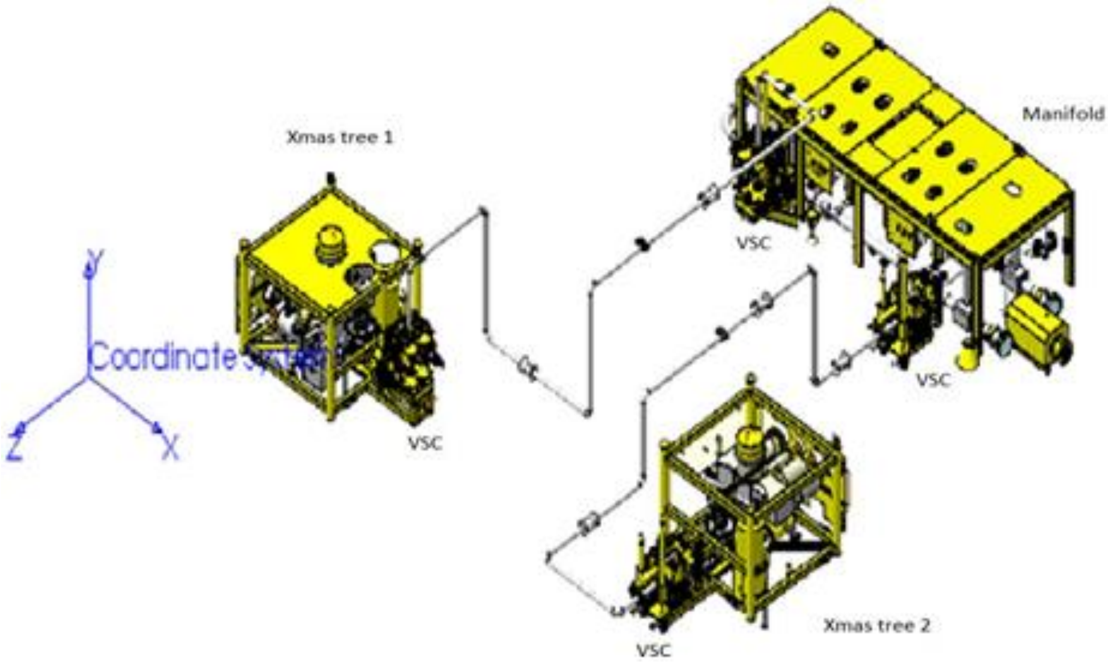


Figure 6:Xmas tree and manifold to be setup in lab

The VSC and the manifold are identical. The VSC is as shown in the figure.

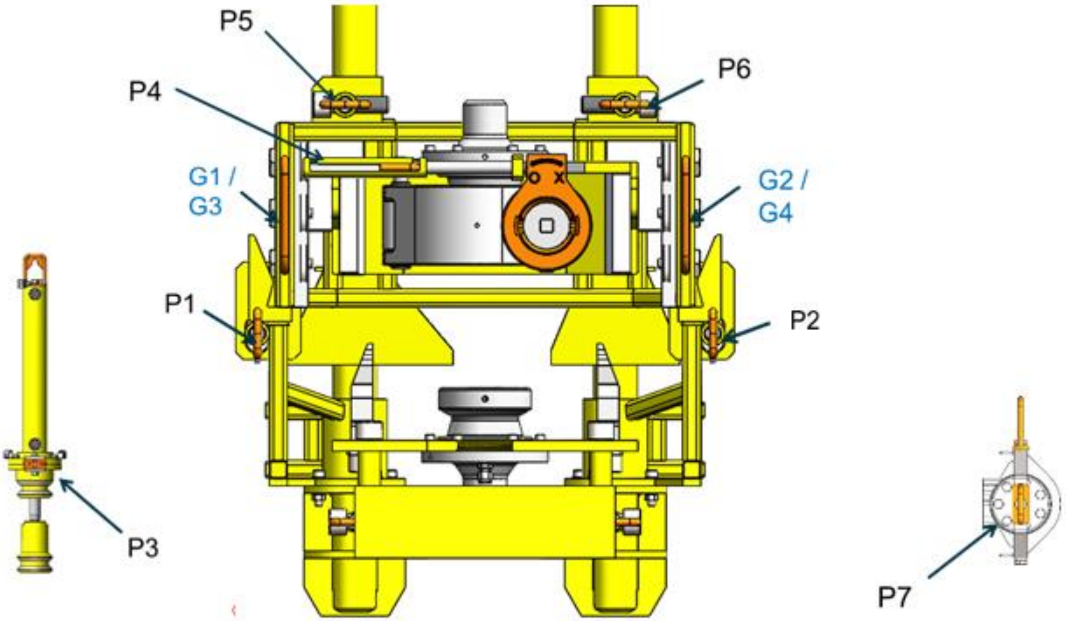


Figure 7: VSC to be mounted on manifold and Xmas tree

3.4 Arms Specification

· Two arms placed on the front of the ROV.

· Left arm will be used as grab arm while right arm will be used as work arm.

· Both arms will not be operated simultaneously.

· The arms will have a visual feedback on screen as well as digital position feedback.

3.5 ROV Operation steps

STEP 1: The ROV will be lowered into the tank using a crane.

STEP 2: The ROV is to reach Xmas tree 1 using navigation system.

STEP 3: The ROV will position itself in front of VSC and stabilize.

STEP 4: Grab arm will be actuated. The grab arm will grab at the provided grab bars G1 or G2 as per convenience.

STEP 5: The work arm will be extended to P1 first. The pin is to be rotated 90° in clockwise direction about positive x axis and then pushed 55 mm along x axis in negative direction. The pin is to be rotated back by 90° in counter clockwise direction about x axis.

STEP 6: The work arm will be moved to P2. The pin is to be rotated 90° in counter clockwise direction about positive x axis and then pushed 55 mm along x axis in negative direction. The pin is to be rotated back by 90° in clockwise direction about x axis.

STEP 7: The work arm will be moved to P3. P3 is a Stroke Tool Handle. A fine adjustment is to be done on this pin.

STEP 8: The work arm will be moved to P4. The pin is to be translated along negative direction in Z axis by 142 mm.

STEP 9: The work arm will be moved to P5. The pin is to be rotated 90° in clockwise direction about positive x axis and then pulled 55 mm along x axis in positive direction. The pin is to be rotated back by 90° in counter clockwise direction about x axis.

STEP 10: The work arm will be moved to P6. The pin is to be rotated 90° in counter clockwise direction about positive x axis and then pulled 55 mm along x axis in positive direction. The pin is to be rotated back by 90° in clockwise direction about x axis.

STEP 11: The work arm will be extended to P7. P7 is stroke tool handle. Minor adjustments are to be done on this pin.

STEP 12: The work arm will be retracted followed by retraction of grab arm.

STEP 13: The ROV is to move to VSC 1 on manifold.

STEP 14: Steps 4 to 12 are to be repeated in same order.

STEP 15: The ROV will move back to the skid from where it will be recoverd.

4.Kinematic Calculations

4.1 Introduction

Robotic arms are one of the most used types of robotic systems in today's world. There are several types of robots, and they can be classified based on factors like the type of joints, number of degrees of freedom, actuation mechanism, etc. Robotic arms can have either revolute joints or prismatic joints. Some can also have a combination of both these types. Some of the most common types of robotics arms are:

1. Cartesian Robot
2. SCARA Robot
3. Articulated Robot
4. Cylindrical Robot
5. Polar Robot
6. Delta Robot

The actuation systems for a robot determines the responses and other characteristics of a robot. The table below compares various characteristics which are desired in a robot.

Table 1: Robot Actuation comparison

Characteristic	Hydraulic	Pneumatic	Belt Driven	Electronic actuators
Response Time	Slow	Fast	Moderate	Fast
Load Capacity	High	Low	Moderate	Moderate
Complexity of construction	High	High	Moderate	Low
Cost	Very High	High	Moderate	Moderate

It can be inferred from the above table; hydraulics actuated robots have a remarkably high load bearing capacity which is an important characteristic for many industrial and explorational applications. These systems, however, lack in the response time which is a huge disadvantage. To overcome this hurdle faced, a PID control loop can be implemented to reduce the errors dynamically, thus reducing the response time drastically, resulting in a fast system.

Kinematic modelling deals with the robot’s motion irrespective of the forces that act on it. It involves study of its geometry and its positions in space. It is used to study relations between end effector and base along with intermediate links. Kinematics is divided into two sections. First, forward kinematics or direct kinematics which gives the cartesian position of the end effector for a given set of fully defined joint parameters. second, inverse kinematics (IK)- which involves calculating joint angles, position for a given end effector position and orientation. IK is more complicated and requires more constraints compared to forward kinematics.

4.2. Arm Setup

The robotic platform is a 6 DOF robotic arm manipulator. It is a serial manipulator having all joints as revolute. The arm geometrical configuration is made up of Waist (Body), Shoulder, Elbow and Wrist in correspondence with the human arm joints. Each of these joints except the wrist has a single DOF. Wrist has 3 DOF (Yaw, Pitch, Roll), thus making the end effector more flexible in terms of operation. The arm is fully actuated with each DOF achieved by a Proportional Derivative Control Valves (PDCVs) with all PDCVs connected to PID. The end-effector is a two-state gripper. The built-in mechanical safety limits restrict the joint motion in case something in the algorithm goes wrong.

Table 2:Arm Specifications

Feature	Description
Position Precision	±1.0mm (approx.)
Operating Movement Speed	20 mm/sec
Hydraulic Pressure	Actuators (70 bar) Gripper (140 bar)
Pump Capacity	80 lpm
Overall Weight	30 Kgs
Load Capacity	15kgs
Hydraulic Hose Diameter	3/8”

The final model of our robot was designed on Solid works software after considering the required specifications. The CAD model is shown below:

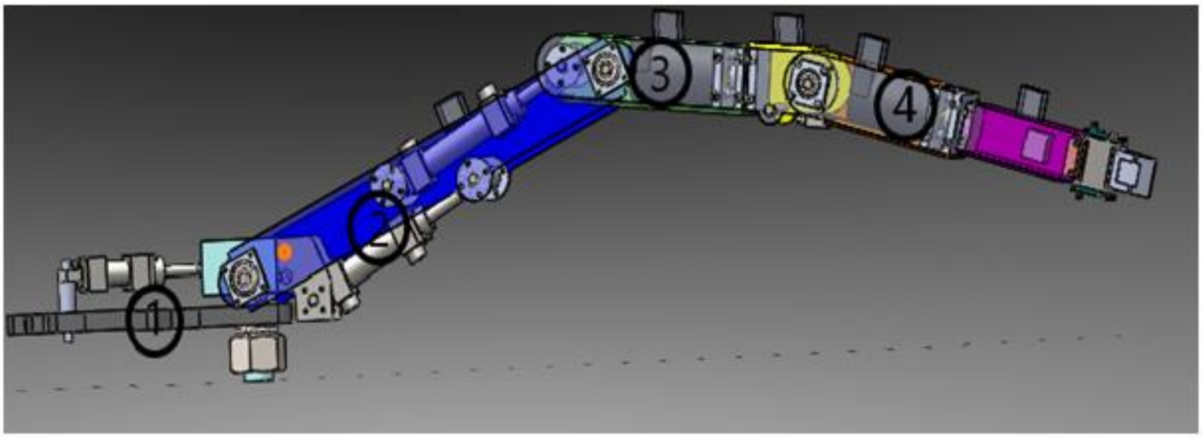


Figure 8:6DOF Arm CAD Model

1 – Waist *2 – Shoulder* *3 – Elbow* *4 – Wrist*

4.3 Kinematic Modelling

Forward kinematics is simple to implement compared to inverse kinematics where some form of numerical method or algorithm is needed to get the desired result. The output of forward kinematics is singular and unique while inverse kinematics can have multiple outputs based upon the constraints.

4.3.1. Forward Kinematic Model

The study of the Kinematic Problem of a robot can be carried out by different methods. The method used here is based on Denavit-Hartenberg (DH) parameters [3]. This Method is systematic in nature and application of method is easy for modelling serial manipulators. DH method is used to develop the kinematic model of the robot because of its versatility and acceptability for modelling any number of joints and links of an arm irrespective of its complexity.

The kinematic diagram for our robot considering the DH parameters (from Table) has been shown below where the axes of rotation for each joint have been considered as the Z axis.

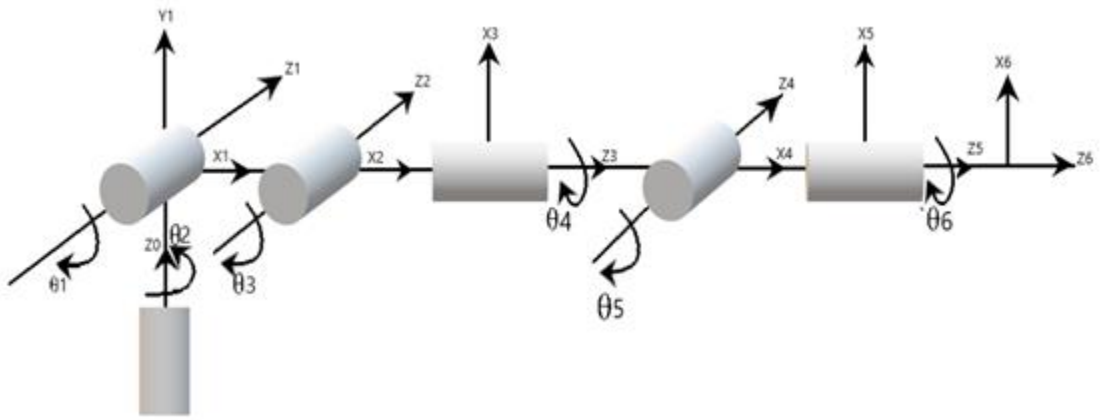


Figure 9:6 DOF arm Kinematic model

DH works with quadruple {αi-1, ai-1, d, θi} [3] represents twist angle, link length, link offset and joint angle, respectively.

Table 3:6 DOF DH parameters

Theta	d (mm)	Alpha (deg)	a (mm)
θ1	37	π/2	0

θ2	0	0	500
θ3	0	π/2	0
θ4	250	-π/2	0
θ5	0	π/2	0
θ6	400	0	0

Using the general form of transformation matrix for each link is given by :

$$T = \left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} \cos\theta & -\sin\theta\cos\alpha & \sin\theta\sin\alpha & r\cos\theta \\ \sin\theta & \cos\theta\cos\alpha & -\cos\theta\sin\alpha & r\sin\theta \\ 0 & \sin\alpha & \cos\alpha & d \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

The Calculation of these individual matrices is direct. Computing each of the link transformation matrix using above equation, as follows

```
A1 =

[ C1, 0,  S1,         0]
[ S1, 0, -C1,         0]
[  0, 1,   0, 37/1000]
[  0, 0,   0,         1]
```

```
A2 =

[ C2, -S2, 0, C2/2]
[ S2,  C2, 0, S2/2]
[  0,   0, 1,    0]
[  0,   0, 0,    1]
```

```
A3 =

[ C3, 0,  S3, 0]
[ S3, 0, -C3, 0]
[  0, 1,   0, 0]
[  0, 0,   0, 1]
```

```
A4 =

[ C4,  0, -S4,   0]
[ S4,  0,  C4,   0]
[  0, -1,   0, 1/4]
[  0,  0,   0,   1]
```

```
A5 =

[ C5, 0,  S5, 0]
[ S5, 0, -C5, 0]
[  0, 1,   0, 0]
[  0, 0,   0, 1]
```

```
A6 =

[ C6, -S6, 0,   0]
[ S6,  C6, 0,   0]
[  0,   0, 1, 2/5]
[  0,   0, 0,   1]
```

Multiplying all the matrices together, starting with the first joint all the way up to the end effector. The final T vector will contain the position of the end effector. The R matrix will contain the orientation of the end effector.

$$ {}^{0}_{6}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

r_{11} = C6*(C5*(S1*S4 + C4*(C1*C2*C3 - C1*S2*S3)) - S5*(C1*C2*S3 + C1*C3*S2)) - S6*(S4*(C1*C2*C3 - C1*S2*S3) - C4*S1)

r_{12}= - S6*(C5*(S1*S4 + C4*(C1*C2*C3 - C1*S2*S3)) - S5*(C1*C2*S3 + C1*C3*S2)) - C6*(S4*(C1*C2*C3 - C1*S2*S3) - C4*S1)

r_{13} = C5*(C1*C2*S3 + C1*C3*S2) + S5*(S1*S4 + C4*(C1*C2*C3 - C1*S2*S3))

p_x = (2*C5*(C1*C2*S3 + C1*C3*S2))/5 + (2*S5*(S1*S4 + C4*(C1*C2*C3 - C1*S2*S3)))/5 + (C1*C2)/2+(C1*C2*S3)/4+ (C1*C3*S2)/4

r_{21} = C6*(C5*(C4*(C2*C3*S1 - S1*S2*S3) - C1*S4) - S5*(C2*S1*S3 + C3*S1*S2)) - S6*(S4*(C2*C3*S1 - S1*S2*S3) + C1*C4)

r_{22} = - S6*(C5*(C4*(C2*C3*S1 - S1*S2*S3) - C1*S4) - S5*(C2*S1*S3 + C3*S1*S2)) - C6*(S4*(C2*C3*S1 - S1*S2*S3) + C1*C4)

r_{23} = C5*(C2*S1*S3 + C3*S1*S2) + S5*(C4*(C2*C3*S1 - S1*S2*S3) - C1*S4)

p_y= (2*C5*(C2*S1*S3 + C3*S1*S2))/5 + (2*S5*(C4*(C2*C3*S1 - S1*S2*S3) - C1*S4))/5 + (C2*S1)/2 + (C2*S1*S3)/4 + (C3*S1*S2)/4

r_{31} = - C6*(S5*(S2*S3 - C2*C3) - C4*C5*(C2*S3 + C3*S2)) - S4*S6*(C2*S3 + C3*S2)

r_{32} = S6*(S5*(S2*S3 - C2*C3) - C4*C5*(C2*S3 + C3*S2)) - C6*S4*(C2*S3 + C3*S2)

r_{33} = C5*(S2*S3 - C2*C3) + C4*S5*(C2*S3 +C3*S2)

p_z = S2/2 + (S2*S3)/4 - (C2*C3)/4 + (2*C5*(S2*S3 - C2*C3))/5 + (2*C4*S5*(C2*S3 + C3*S2))/5 + 37/1000

4.3.2 Inverse Kinematic Model

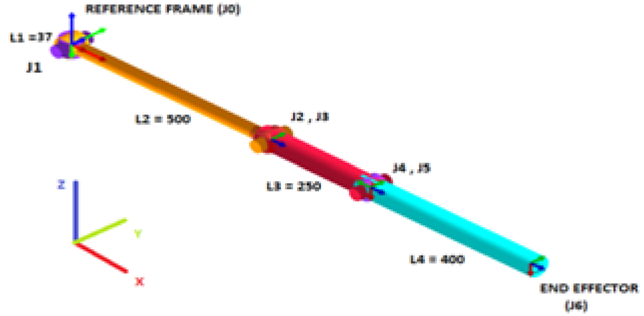


Figure 10: 6DOF Inverse Kinematic Model

Inverse kinematics involves the calculation of joint angles with the end effector coordinates as input. This is useful in applications where we know the final position which is to be reached with respect to a reference frame. The inverse kinematic gives the joints values i.e.(θ1, θ2, θ3, θ4, θ5, θ6), where theta is the angle to rotate around the Z axis.Multiple solution can be obtained for one end effector position, though not all solutions are feasible, the solution that provides a smooth operation and minimum joint motion is said to be best solution. Optimal Solution to supply a robust operation. Software such as MATLAB directly provides the optimal solution.

In this paper, both analytical methods and software approaches are mentioned. This method verifies that for any object in the vicinity of the arms, the model provides correct joint angles.

Inverse Kinematic calculations for 1st three joints:

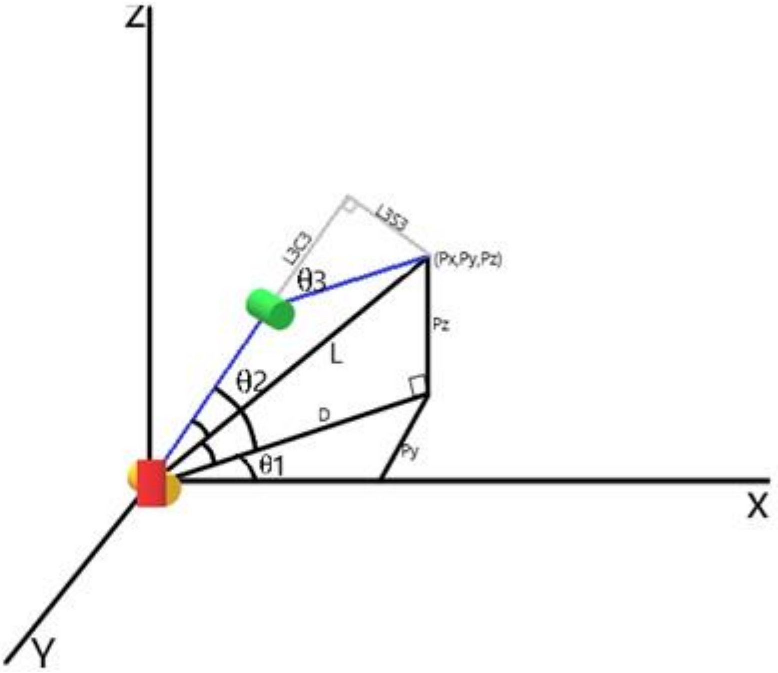


Figure 11: Inverse Kinematics coordinate system

The equations for inverse kinematics of the first three joints are as follows:

Θ_1 =arctan(Py/Px)

-(1)

Θ_3 = arctan($(1-M^2)^{½}$ /M)

-(2)

Where, M = $(L^2 - L_2^2 - L_3^2)/ 2L_2L_3$

Θ_2 = arctan (L_3*Sin Θ_3*D + P_z (L_3Cos Θ_3 + L_2)/L_3*Cos Θ_3+L_2*D - L_3 *Sin Θ_3*P_z) -(3)

Where, D = $(P_x^2 + P_y^2)^{½}$

4.3. Verification of kinematic model & MATLAB GUI

The verification of the model is done by the physical model and cross checking the results with analytical calculations. A proposed analytical method is given in inverse modelling. MATLAB is used to make the calculation part simpler. Verification is done based on required accuracy of 1mm for forward kinematics and 1 degrees for inverse kinematics.

4.3.1. Verification of forward kinematic model

The analytical calculations for forward kinematics of the robot have been verified on MATLAB using the robotics toolbox. The analytical calculations given in section 4.2.2.

The same inputs were given in the MATLAB environment and the position of the robot was plotted after running the MATLAB script. The code also gives the end effector positions as an output. These coordinates are compared with the ones obtained from analytical calculations and have been compared in the table below.

Table 4: Kinematics verification result

Input angles	Θi (deg)	End effector Position(mm)	MATLAB Output(mm)	Experimental Output(mm)
θ1	0	X	250	250

θ2	90	Y	-1	0
θ3	0	Z	137	137
θ4	0			
θ5	-90			
θ6	0			

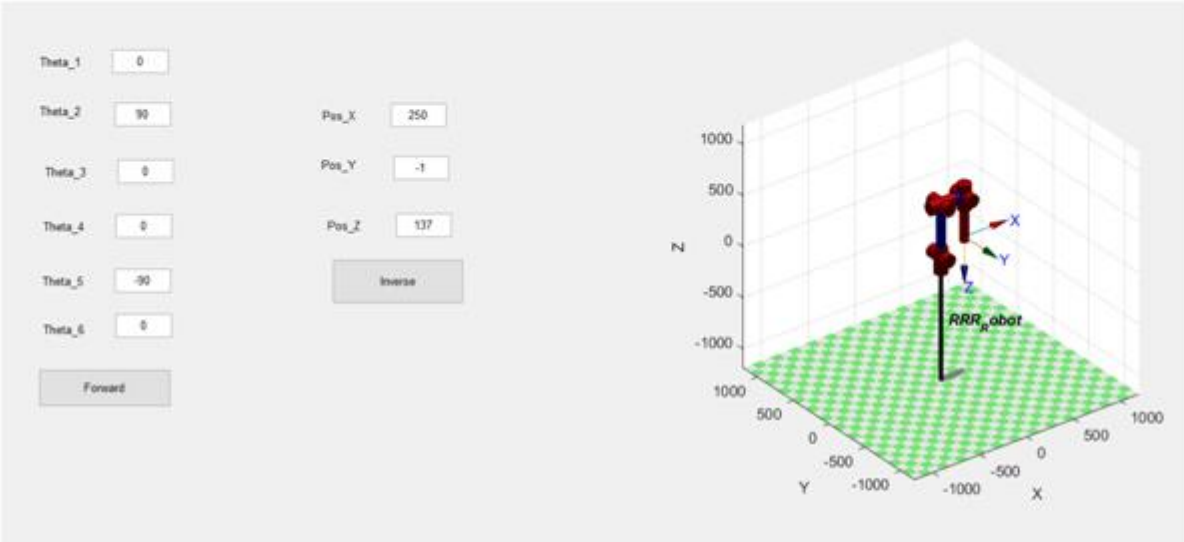


Figure 12: Matlab GUI for forward and inverse kinematics

4.3.2. Verification of Inverse Kinematic Model

The verification of inverse kinematic model involves the measurement of joint angles for a particular end effector position using a set of potentiometers mounted on the joint axis. Potentiometers are sensors which convert the rotational motion into a change of resistance. This change of resistance can be converted into an electrical signal and this feedback would be used to perform the inverse kinematic calculated analytically. The Accuracy of potentiometer readings is 0.1 deg. We have used the robotics toolbox mentioned earlier to calculate the joint angles from a desired end effector position. These values have been further verified by analytical calculations and then compared with the physical potentiometer feedbacks mentioned above.

The results from analytical calculations are compared with the actual end effector position.

The table below summarizes the joint angles obtained for the end effector coordinates (500, -100,100).

Joint Number	Parameter	Potentiometer	MATLAB
1	θ1	-11.6	-12
2	θ2	79.30	80
3	θ3	-9.80	-10
4	θ4	0	0
F5	θ5	-44.2	-44
6	θ6	0.1	0

4.3.3 MATLAB GUI for verification

S Mahalingam. has discussed the method and algorithm for the development of a simple Graphic User Interface (GUI) which can be used for performing the forward and inverse kinematics calculations. We have modified and developed a GUI for simulation of our robot. We have re-developed by considering the DH parameters for our robot. This GUI will be developed further in the coming months to make it a more polished interface. We incorporated the real time angle feedback from the potentiometer feedback system.

The backend code for the GUI is as follows:

```
function varargout = InverseKin6dof(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @InverseKin6dof_OpeningFcn, ...
                   'gui_OutputFcn',  @InverseKin6dof_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function InverseKin6dof_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
function varargout = InverseKin6dof_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function Theta_1_Callback(hObject, eventdata, handles)
function Theta_1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Theta_2_Callback(hObject, eventdata, handles)
function Theta_2_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function Theta_3_Callback(hObject, eventdata, handles)

function Theta_3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Theta_4_Callback(hObject, eventdata, handles)
function Theta_4_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function Theta_5_Callback(hObject, eventdata, handles)

function Theta_5_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function Theta_6_Callback(hObject, eventdata, handles)
function Theta_6_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Pos_X_Callback(hObject, eventdata, handles)
function Pos_X_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Pos_Y_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function Pos_Y_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function Pos_Z_Callback(hObject, eventdata, handles)
function Pos_Z_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function btn_Forward_Callback(hObject, eventdata, handles)
Th_1 = str2double(handles.Theta_1.String)*pi/180;
Th_2 = str2double(handles.Theta_2.String)*pi/180;
Th_3 = str2double(handles.Theta_3.String)*pi/180;
Th_4 = str2double(handles.Theta_4.String)*pi/180;
Th_5 = str2double(handles.Theta_5.String)*pi/180;
Th_6 = str2double(handles.Theta_6.String)*pi/180;

L_1 = 37;
L_2 = 500;
L_3 = 0;
L_4 = 250;
L_5 = 0;
L_6 = 400;

L(1) = Link([0 L_1 0 pi/2]);
L(2) = Link([0 0 L_2 0]);
L(3) = Link([0 0 L_3 pi/2]);
L(4) = Link([0 L_4 0 -pi/2]);
L(5) = Link([0 0 L_5 pi/2]);
L(6) = Link([0 L_6 0 0]);
Robot = SerialLink(L);
Robot.name = 'RRR_Robot';

Robot.plot([Th_1 Th_2 Th_3 Th_4 Th_5 Th_6]);
T = Robot.fkine([Th_1 Th_2 Th_3 Th_4 Th_5 Th_6]);
handles.Pos_X.String = num2str(floor(T.t(1)));
handles.Pos_Y.String = num2str(floor(T.t(2)));
handles.Pos_Z.String = num2str(floor(T.t(3)));


% --- Executes on button press in btn_Inverse.
function btn_Inverse_Callback(hObject, eventdata, handles)
PX = str2double(handles.Pos_X.String);
PY = str2double(handles.Pos_Y.String);
PZ = str2double(handles.Pos_Z.String);
L_1 = 37;
L_2 = 500;
L_3 = 0;
L_4 = 250;
L_5 = 0;
L_6 = 400;
```

```
L(1) = Link([0 L_1 0 pi/2]);
L(2) = Link([0 0 L_2 0]);
L(3) = Link([0 0 L_3 pi/2]);
L(4) = Link([0 L_4 0 -pi/2]);
L(5) = Link([0 0 L_5 pi/2]);
L(6) = Link([0 L_6 0 0]);
Robot = SerialLink(L);
Robot.name = 'RRR_Robot';
T = [ 1 0 0 PX;
 0 1 0 PY;
 0 0 1 PZ;
 0 0 0 1];
J = Robot.ikine(T,[0 0 0],'mask',[1 1 1 0 0 0])*180/pi;
handles.Theta_1.String = num2str(floor(J(1)));
handles.Theta_2.String = num2str(floor(J(2)));
handles.Theta_3.String = num2str(floor(J(3)));
handles.Theta_4.String = num2str(floor(J(4)));
handles.Theta_5.String = num2str(floor(J(5)));
handles.Theta_6.String = num2str(floor(J(6)));
Robot.plot(J*pi/180);
```

4.3.4 Converting joint angles into equivalent linear displacement of hydraulic cylinders

There are various methods to transfer joint angles into linear displacement of hydraulic cylinders based on geometry, differential calculus. We discuss the use of cosine rule to transfer required angle into required cylinder extension position. As the length of every link is fixed there are only two variables that are present into the formed triangle. The length of cylinder (fixed length + stroke length) and required angle α (required angle). Implantation of cosine rule can be done easily in primitive controllers while the implantation of counterpart differentiation equations requires higher computing power.

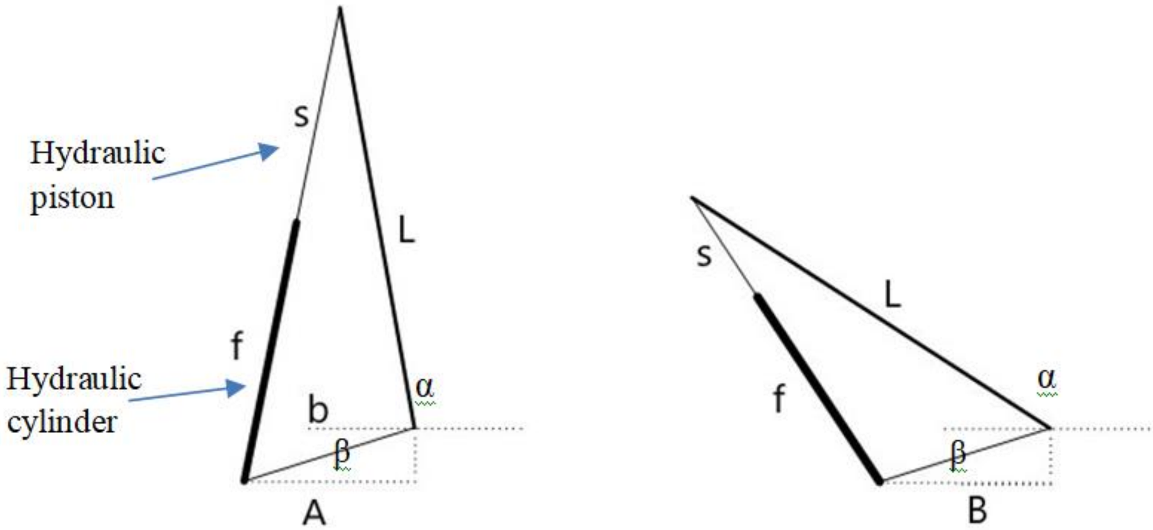


Figure 13: Line diagram of hydraulic cylinder and joint angle triangle

Fig A- cylinder in extended position

Fig B- cylinder in intermediate position α- required link angle with previous link

L- link length

s- Stroke length of piston

f- Fixed length of cylinder

b- fixed distance between link and cylinder

According to Cosine Rule: -

Thus

The equation (5) gives the required stroke length of cylinder to achieve the desired angle of rotation. The parameters in the equation can be changed to suit various types of actuators of different types of robots. Dynamic modelling based upon the above formula is also possible.

5.PID Control for Arms:

Studies have shown that 90% of all controllers in the process industry have a PID structure. PID loops can handle a wide range of control problems and are popular due to their simple configuration and off-the-shelf tuning [8].

A PID controller, has three principal control actions, i.e., the proportional action, integral action, and the derivative action. All these control actions are summed up together to obtain a single control effort [9].

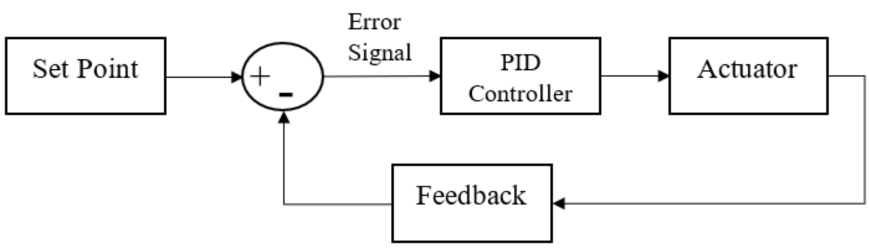


Figure 14:PID Control Loop

5.1Design and simulation of PID controllers for the arms.

A proportional–integral–derivative (PID) controller is a control loop mechanism employing feedback that is widely used in industrial control systems. A PID controller continuously calculates an error value e(t) as the difference between the desired setpoint (SP) and a measured process variable (PV) and applies a correction based on proportional, integral, and derivative gains.

PID equation:

The 4DOF and the 6DOF arms consist of 4 and 6 revolute joints respectively. Each joint will have an associated PID controller. In this setup, one PID loop is executed at a time, hence link-wise PID control can be done.

During the practical implementation of the PID controllers

- The desired setpoint is received from the workstation through the inverse kinematics algorithm.
- The current angle of each of the links will be given by the potentiometers.
- The individual PID controllers will be implemented on the Arduino Mega for Arms.
- The proportional (kp), integral (ki), and derivative (kd), gains determined from the Simulink model will be utilized for these controllers.

Algorithm for PID Control

- Current potentiometer values i.e., joint angles are obtained.
- IK algorithm running at the workstation computes final required values and sends them to Nano Pi M4.
- With new values of the potentiometer, the PID controller calculates output for each link.
- PDCVs are operated according to this control signal and new values of potentiometers are obtained and the process is repeated.

5.1.1 Modelling of the PID controller on Simulink

The steps to create a model from the CAD model of the arms are:

a. All the body properties such as mass, the moment of inertia, or the position of the central center of gravity are exported from a CAD/CAM environment.
b. The manipulator CAD model is then to be imported to MATLAB Simulink as an XML file with the aid of the Simulink Multibody Link Toolbox.
c. The actual Simulink model of the arm was very detailed and hence the redundant parts of the arm have been removed and the new CAD model containing only the necessary parts as shown in the model rendered below has been imported into Simulink using the ‘Simulink Multibody Link Toolbox’. Simulink Multibody Link Toolbox is a modeling environment for the design

and simulation of rigid multibody machines and their motions, using the standard Newtonian dynamics of forces and torques.

d. The following parts are kept in the model:
    i. Ground: It grounds one side of a joint to a fixed location in the world coordinate system.
    ii. Environment: It defines the simulation environment parameters like gravity.
    iii. Weld: It is a zero degree of freedom joint.
    iv. Body: It represents the different rigid bodies that are a part of the CAD model.
    v. Revolute Joints: These are the joints of the arm that are to be actuated.

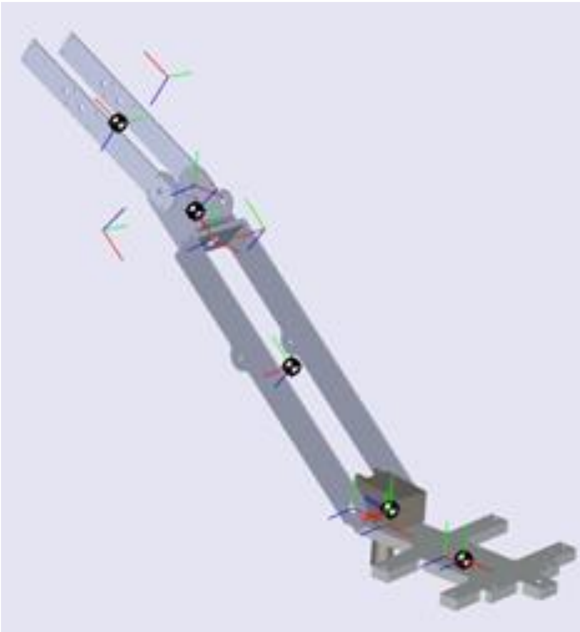


Figure 15: Visualization of 4DOF Arms in Simulink

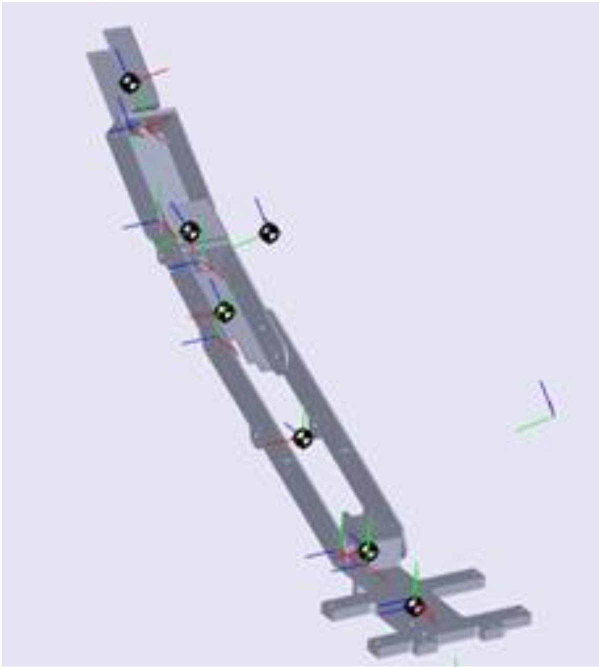


Figure 16:Visualization of 6DOF Arms in Simulink

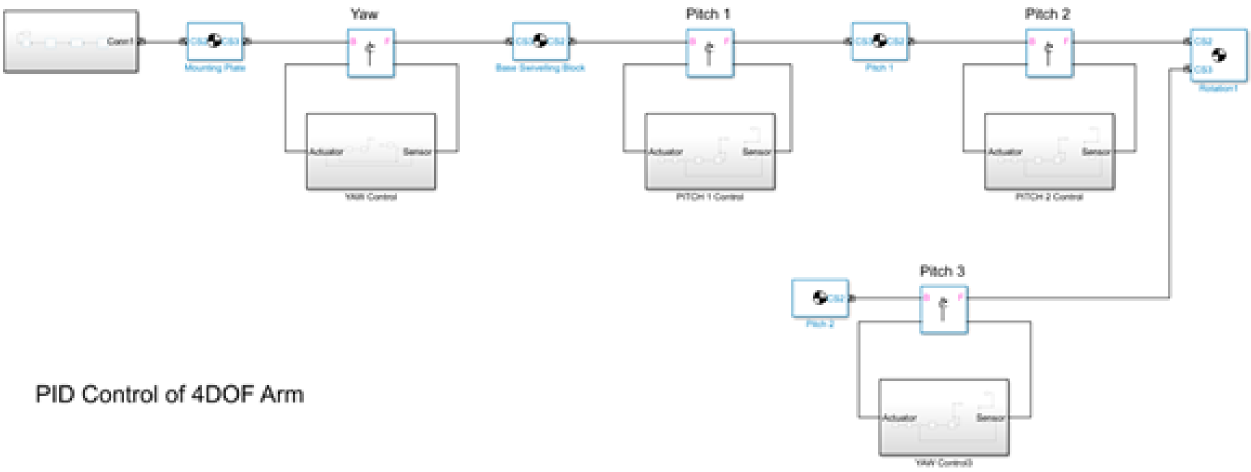


Figure 17: Simulink Model for the PID Control of 4DOF arm

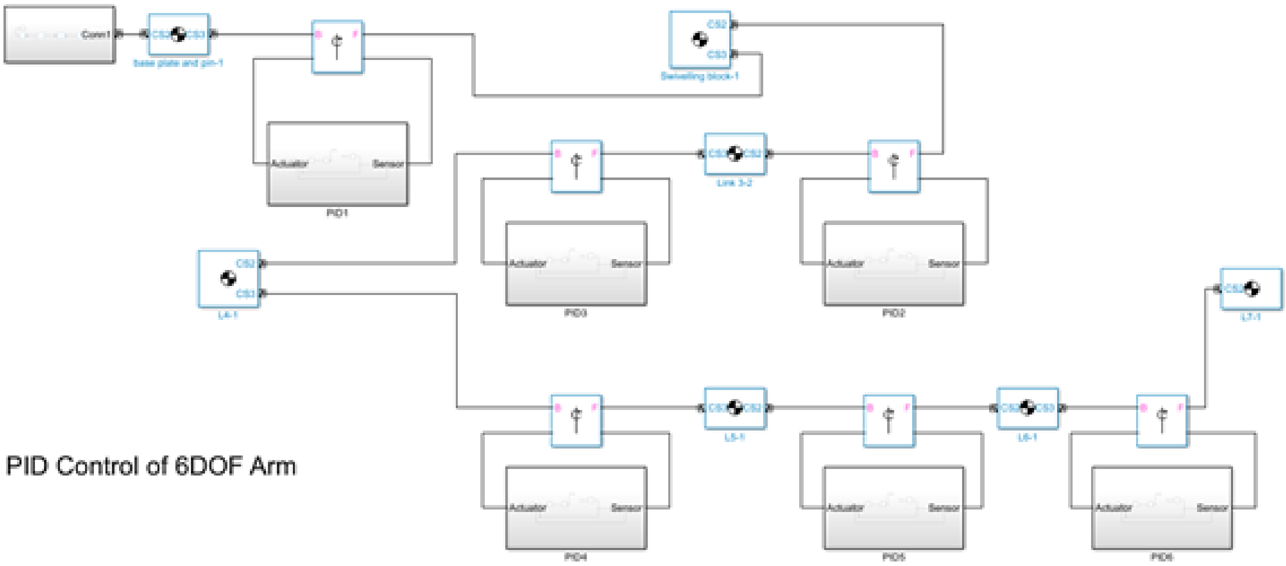


Figure 18: Simulink Model for the PID Control of 6DOF arm

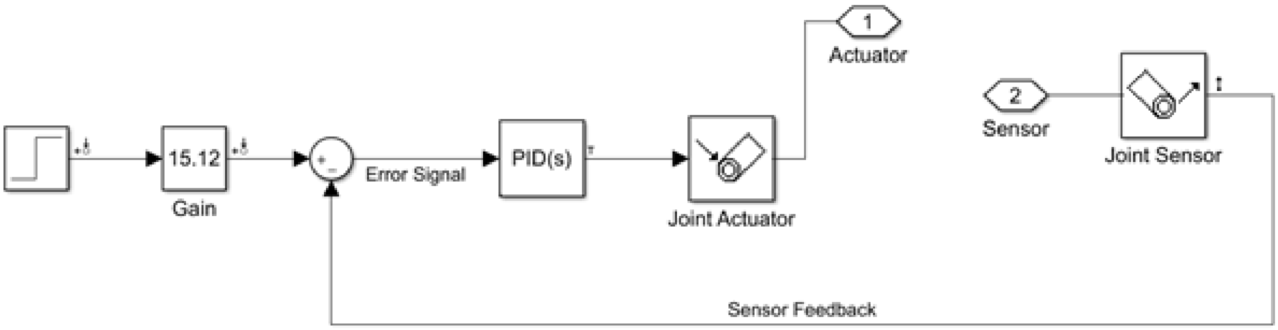


Figure 19: PID Controller Subsystem

5.2Tuning the PID Controller on Simulink

1. Once the model has been generated using the Simscape Multibody link, the PID block needs to be tuned to obtain optimum performance.
2. The PID tuner can be used to tune the controller for optimum performance. The PID tuner offers an option to tune the controller as per our requirements. The parameters of the PID controller response can be observed in the tuner. Parameters like the PID gains (Kp, Ki, Kd), rise time, overshoot, settling time, steady-state error are obtained.
3. This particular project has the requirement of accuracy more than the speed of operation; hence the controller has been tuned in a way that the PID response has

moderate rise time, a faster settling time, and the lowest possible overshoot and steady-state error.

4. Once the tuning has been done for one PID subsystem, this process was repeated for all the PID controllers of the 4DOF and the 6DOF arm for a total of 10 controllers.
5. The PID parameters obtained from the tuner are:
   a) P – 197.3
   b) I – 67067
   c) D – 0.14
6. So, we can see that the value for derivative component is almost zero. Hence, it is essentially following a PI control.

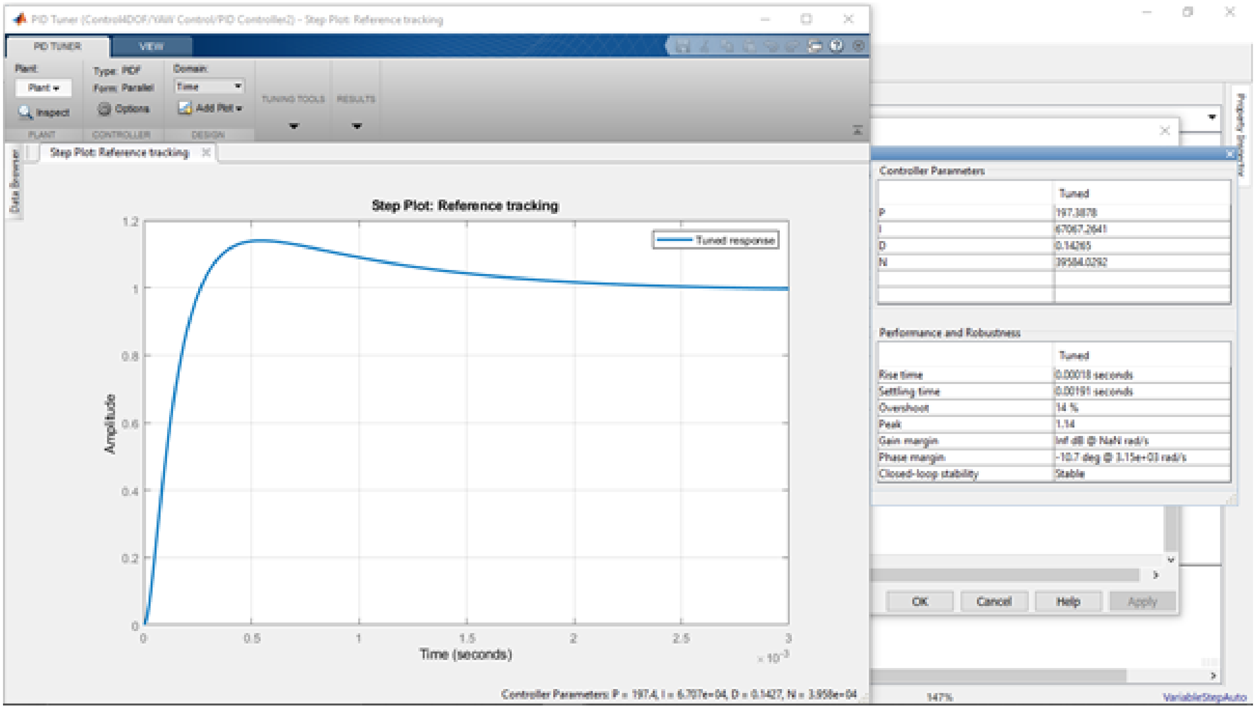


Figure 20: PID Tuning and Response of PID Controller

5.3 Implementation of PID Control on ROV

The PID gain values obtained from the model simulated on MATLAB are utilized to implement PID control on Arduino.

Initially, the PID library available on Arduino was going to be used for this implementation but after preliminary testing, it was observed that this wasn't ideal for our application. Drawbacks of the Arduino PID library were as follows:

1. This library was intended to be used for applications that required a single control signal for their operation. As the PDCVs require two operating signals -forward and reverse, accommodating this requirement was hard.
2. The library consisted of many redundant parts that were not required and hence could have potentially slowed down the processing when multiple PID loops were executed simultaneously.

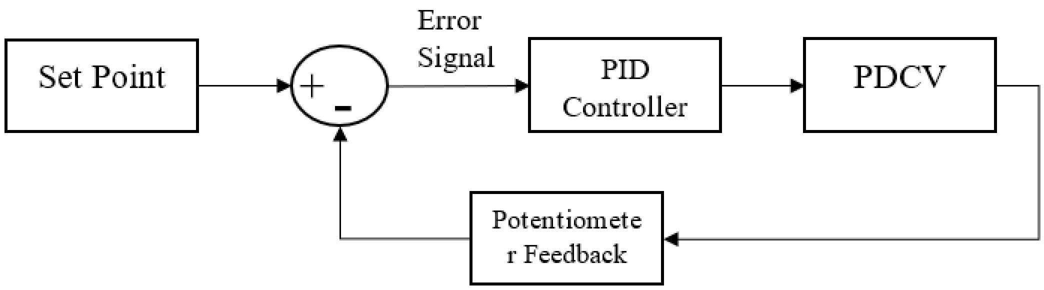
In order to counter these problems, a separate library was created by us which was specifically designed for our application and was efficient.

This library consists of a simpler code and was suitable for producing the control signals in the form of two varying PWM signals for the motion of the arm.

Testing and Results

- The PID control was tested with two links of the 4DOF arm that had potentiometers mounted.
  The results are as follows-
  Observed time to settle-20 sec
  Steady state error- 2 deg

Link	Setpoint	Angle Attained with PID Control	Attainment of Setpoint	Time taken to reach Setpoint (seconds)	Observation
4DOF_Link_1	70	69	Yes	8.90	Took longer to settle as it was oscillating significantly due to a mechanical problem.
4DOF_Link_2	15	15	Yes	2.45	Stabilized at setpoint with minor oscillations.



Set Point
+
-
Error
Signal
PID
Controller
PDCV
Potentiomete
r Feedback

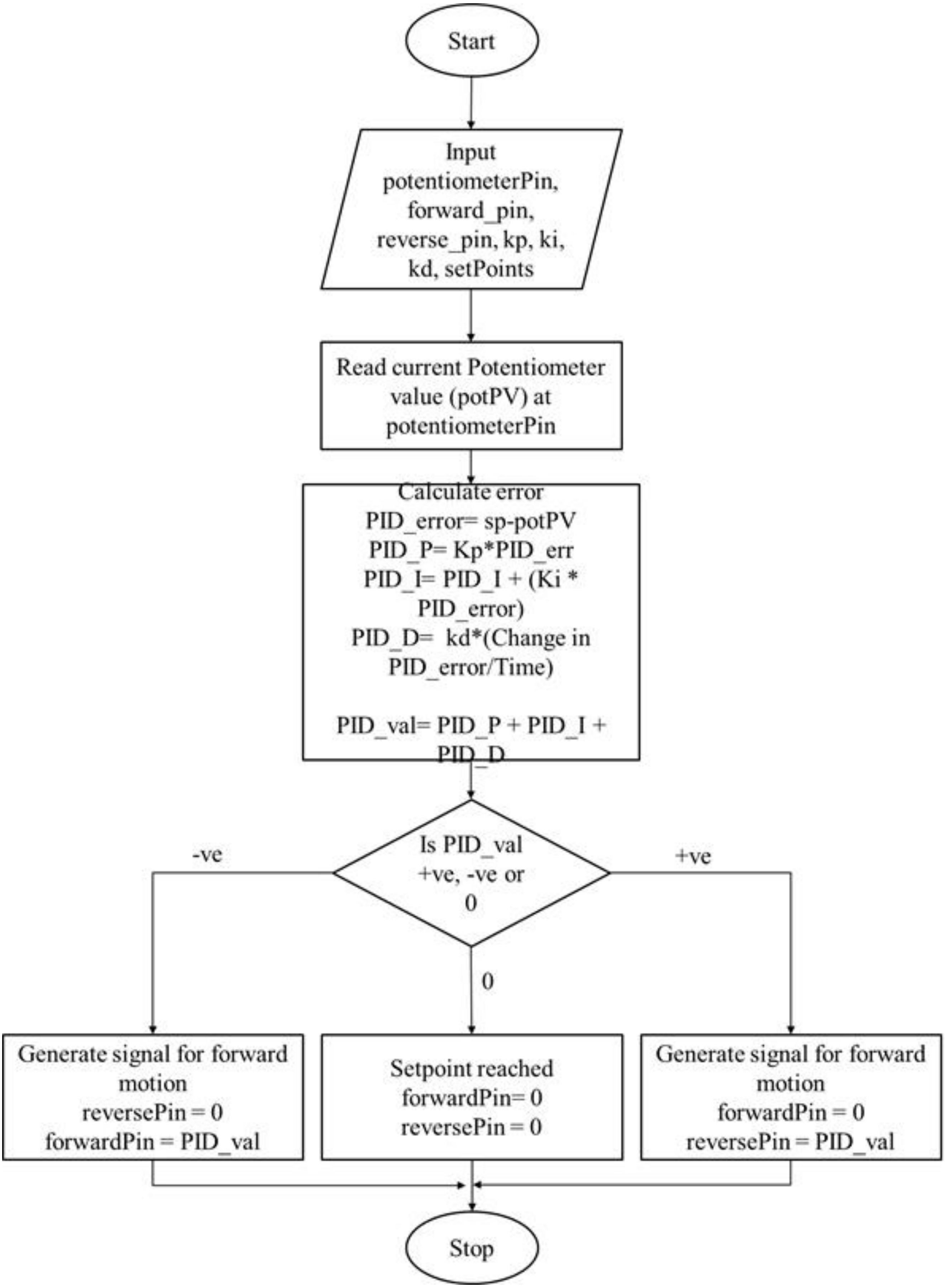

Start
Input
potentiometerPin,
forward_pin,
reverse_pin, kp, ki,
kd, setPoints
Read current Potentiometer
value (potPV) at
potentiometerPin
Calculate error
PID_error= sp-potPV
PID_P= Kp*PID_err
PID_I= PID_I + (Ki *
PID_error)
PID_D= kd*(Change in
PID_error/Time)
PID_val= PID_P + PID_I +
PID_D
Is PID_val
+ve, -ve or
0
-ve
+ve
0
Generate signal for forward
motion
reversePin = 0
forwardPin = PID_val
Setpoint reached
forwardPin= 0
reversePin = 0
Generate signal for forward
motion
forwardPin = 0
reversePin = PID_val
Stop

Figure 21: Flowchart of PID Control Loop

5.4 MATLAB Code for Graphical Outputs

5.4.1 Forward Kinematics

In this the plot function was used to plot the calculated values of x, y and z co-ordinates from the MATLAB code.

The code was tested for the following values:

Table 5: Forward Kinematics input parameters

Theta	**Degrees**	**Co ordinates**	**mm**
Theta 1	60	X	722
Theta 2	20	Y	568.3
Theta 3	30	Z	-35.2
Theta 4	80		
Theta 5	60		
Theta 6	0		

```
%defining the link lengths
L_1 = 37;
L_2 = 500;
L_3 = 0;
L_4 = 250;
L_5 = 0;
L_6 = 400;

%creating link objects in the MATLAB ecosystem
L(1) = Link([0 L_1 0 pi/2]);
L(2) = Link([0 0 L_2 0]);
L(3) = Link([0 0 L_3 pi/2]);
L(4) = Link([0 L_4 0 -pi/2]);
L(5) = Link([0 0 L_5 pi/2]);
L(6) = Link([0 L_6 0 0]);

Robot = SerialLink(L); %Joining all the links together
qz = [0 0 0 0 0 0]; %defining the initial joint angles
qr = deg2rad([60 20 30 80 60 0]); %defining the final joint angles
t = [0:1:20];   % generate a time vector
q = jtraj(qz, qr, t); % compute the joint coordinate trajectory
K = Robot.fkine(q); %perform forward kinematics
R = K.transl; %extract the cartesian coordinates from the final transformation matrix

%plotting the results
subplot(3,1,1);
a1 = plot(t, R(:,1));
xlabel('Time (s)');
ylabel('X (m)');
subplot(3,1,2);
a2 = plot(t, R(:,2));
xlabel('Time (s)');
ylabel('Y (m)');
subplot(3,1,3);
a3 = plot(t, R(:,3));
xlabel('Time (s)');
ylabel('Z (m)');
grid
hold on
```

5.4.2 Forward Kinematics Plots

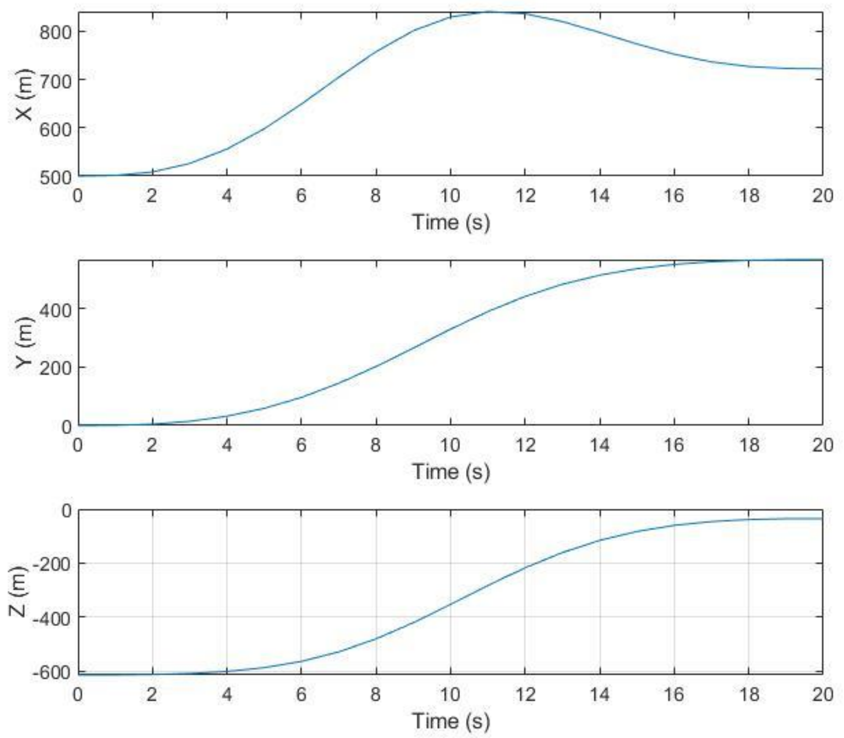


Figure 22: Forward Kinematics plot

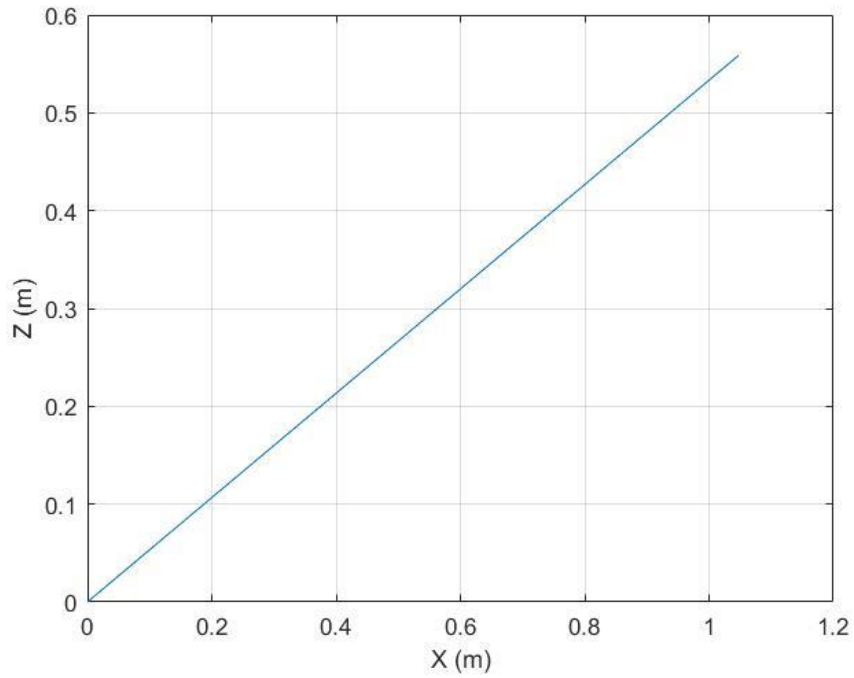


Figure 23: End effector trajectory

5.4.3. Inverse Kinematics

In this the code was run to get the joint angles for the given end effector coordinates.

The values on which the code was tested are:

Coordinates	**mm**	**Theta**	**Degrees**
X1	500	Theta 1	-13.14
Y1	-1	Theta 2	-36.4
Z1	-613	Theta 3	73.96
X2	850	Theta 4	0.81
Y2	-200	Theta 5	15.27
Z2	-700	Theta 6	0

```
%defining the link lengths of robotic arm
L_1 = 37;
L_2 = 500;
L_3 = 0;
L_4 = 250;
L_5 = 0;
L_6 = 400;

%creating link objects in the MATLAB ecosystem
L(1) = Link([0 L_1 0 pi/2]);
L(2) = Link([0 0 L_2 0]);
L(3) = Link([0 0 L_3 pi/2]);
L(4) = Link([0 L_4 0 -pi/2]);
L(5) = Link([0 0 L_5 pi/2]);
L(6) = Link([0 L_6 0 0]);
Robot = SerialLink(L); %Joining all the links together

T1 = transl(500, -1, -613); % define the start point
T2 = transl(850, -200, -700);   %and destination
T = ctraj(T1, T2, 20);  % compute a Cartesian path
q = rad2deg(Robot.ikine(T,'mask',[1 1 1 0 0 0])); %performing inverse kinematics
```

```
%plotting the results
figure(1)
subplot(3,1,1); plot(q(:,1)); xlabel('Time (s)'); ylabel('Joint 1 (deg)');
subplot(3,1,2); plot(q(:,2)); xlabel('Time (s)'); ylabel('Joint 2 (deg)');
subplot(3,1,3); plot(q(:,3)); xlabel('Time (s)'); ylabel('Joint 3 (deg)');

figure(2)
subplot(3,1,1); plot(q(:,4)); xlabel('Time (s)'); ylabel('Joint 4 (rad)');
subplot(3,1,2); plot(q(:,5)); xlabel('Time (s)'); ylabel('Joint 5 (rad)');
subplot(3,1,3); plot(q(:,6)); xlabel('Time (s)'); ylabel('Joint 6 (rad)');
```

5.4.4. Inverse Kinematics Plots

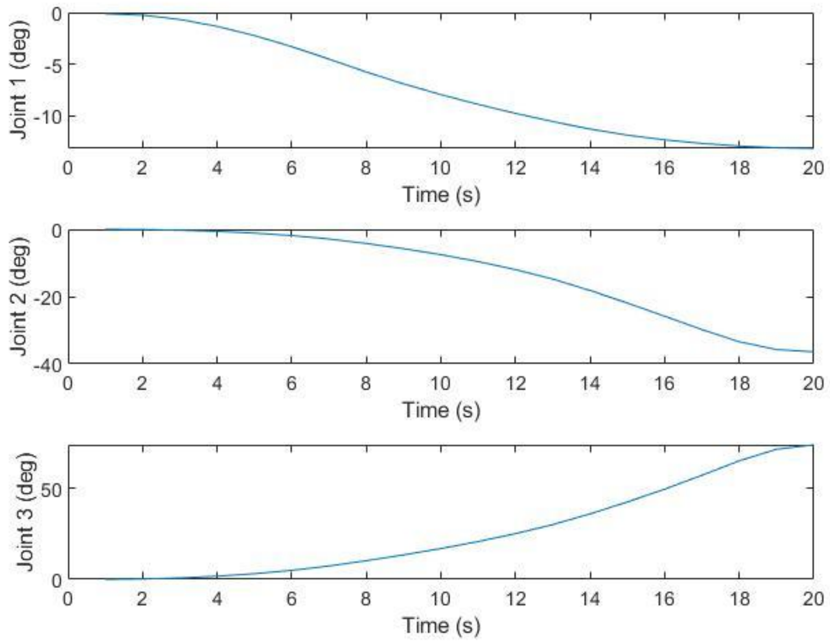


Figure 24: Inverse Kinematics plot(a)

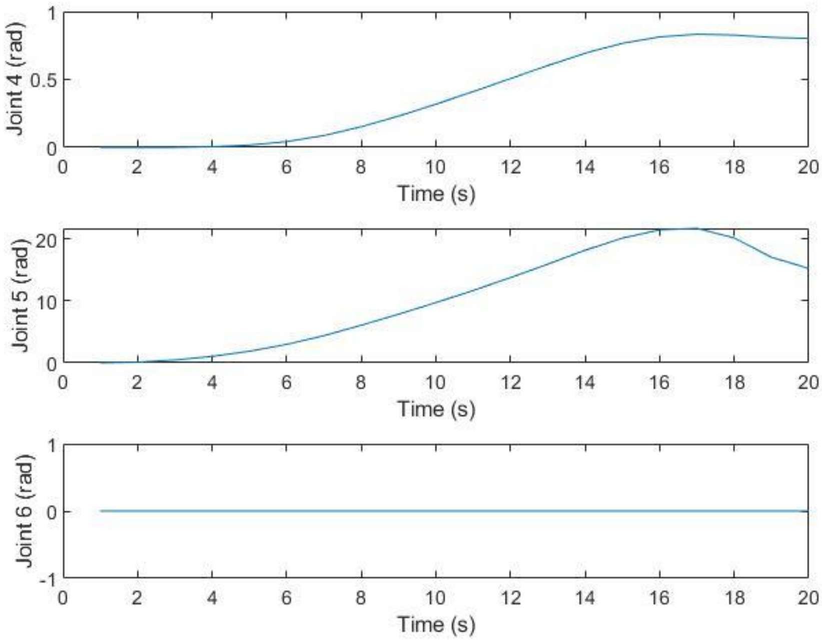


Figure 25: Inverse Kinematics plot(b)

5.5 Voltage mapping of the PDCV

Various speeds of the arms were obtained by varying the control voltage. Based on the requirement of the project moderate to slow speed of the arm movement is most suitable.

The following results were obtained during the testing of the arms:

Table 6: Variation of PDCV Speed with Control Voltage

Control Voltage given to PDCV	Speed
0 to 1V	Dead zone
1V to 2V	Low speed
2V-3.5V	Moderate speed
3.5V-5V	High speed

Table 7: PDCV Control Voltage Vs Angular Velocity

Control Voltage (V)	Angular Velocity (deg/s)
0	0
0.5	0
1	0
1.5	4.12
2	9.31
2.5	18.6
3	31.7
3.5	41.5
4	49.09
4.5	67.5
5	77.14

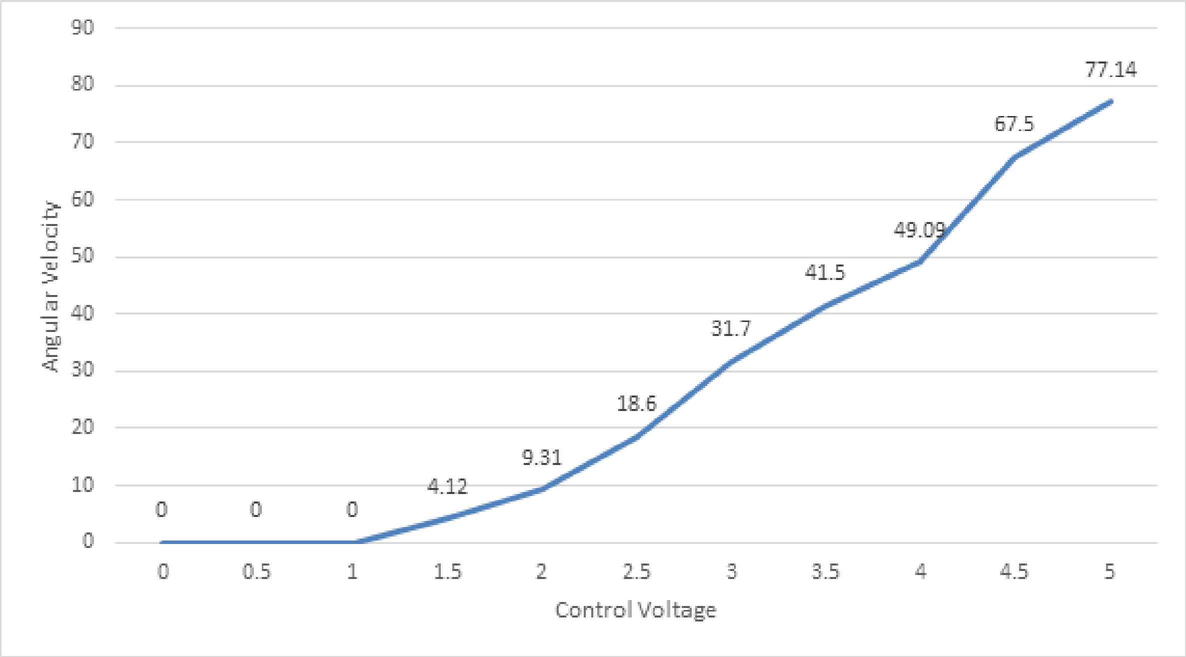


Fig 5.3. PDCV Control Voltage Vs Angular Velocity

6. Mechanical Design and Modifications

6.1 Potentiometer Mountings

6.1.1 4 DOF Potentiometer Mountings CAD and Analysis

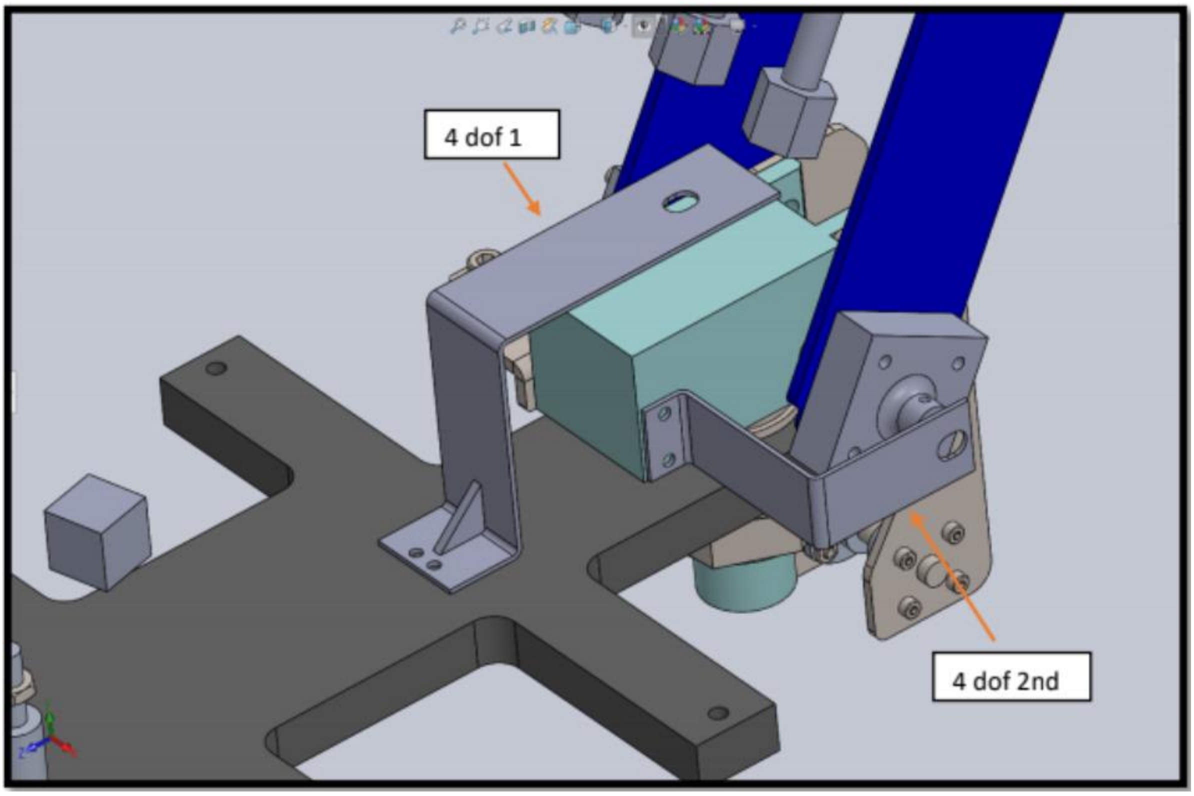
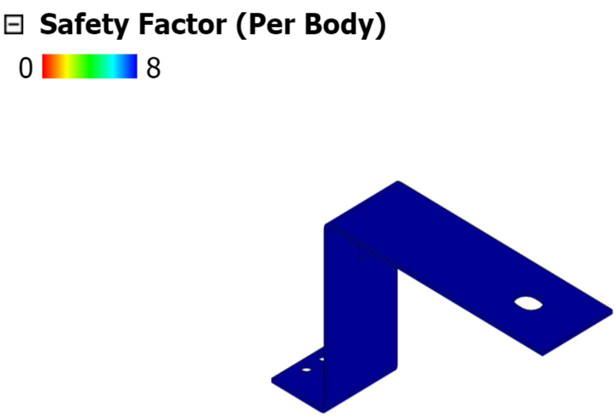
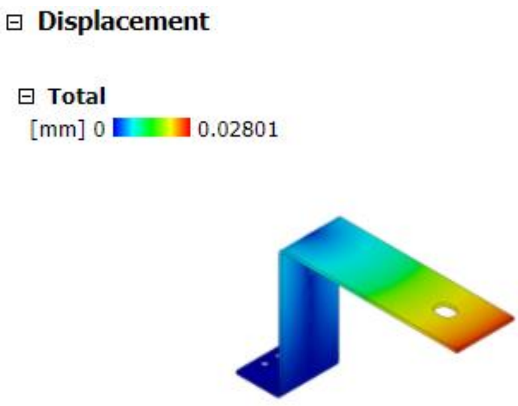


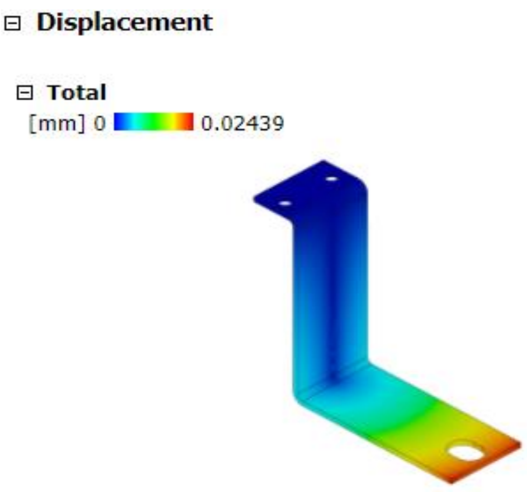
Figure 26:4 DOF Pot mounting position CAD

4 DOF Mount 1 Analysis





4 DOF Mount 2 Analysis



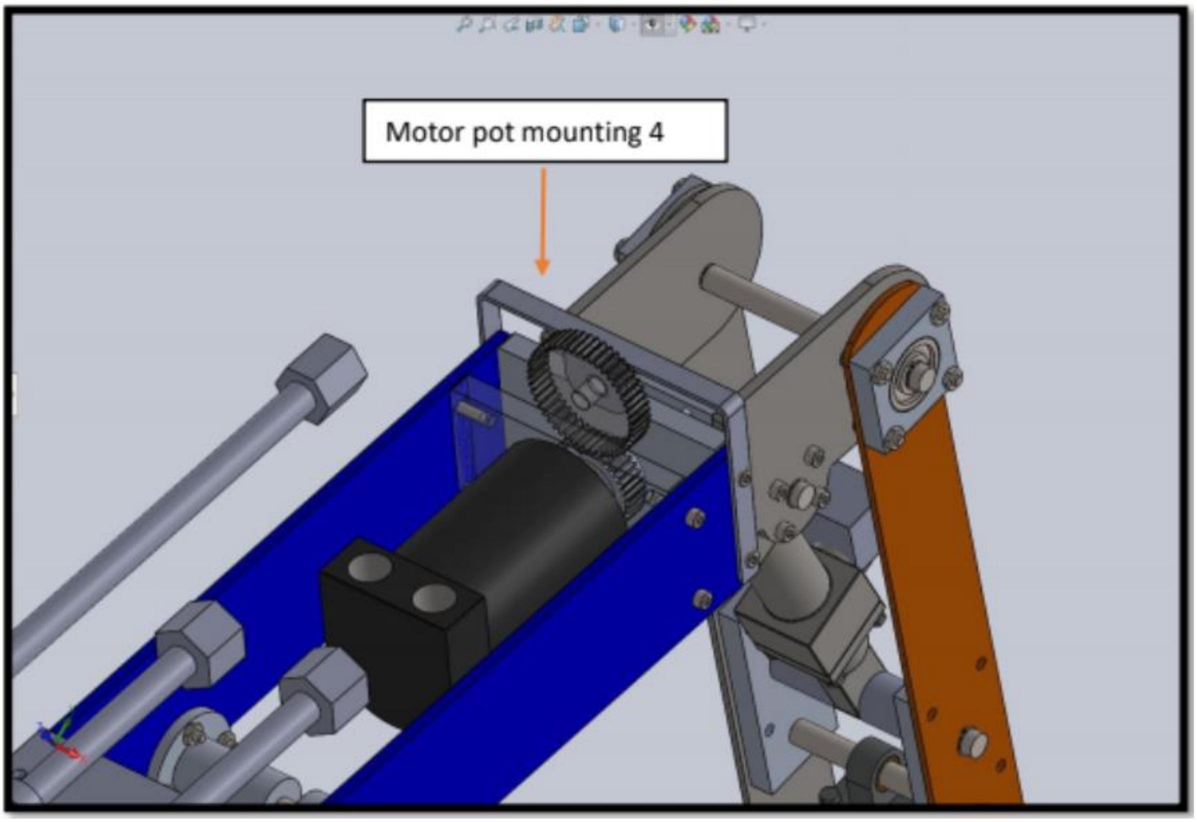


Figure 27: Motor pot Mounting Position CAD

6.1.2. 6 DOF Potentiometer Mounting CAD

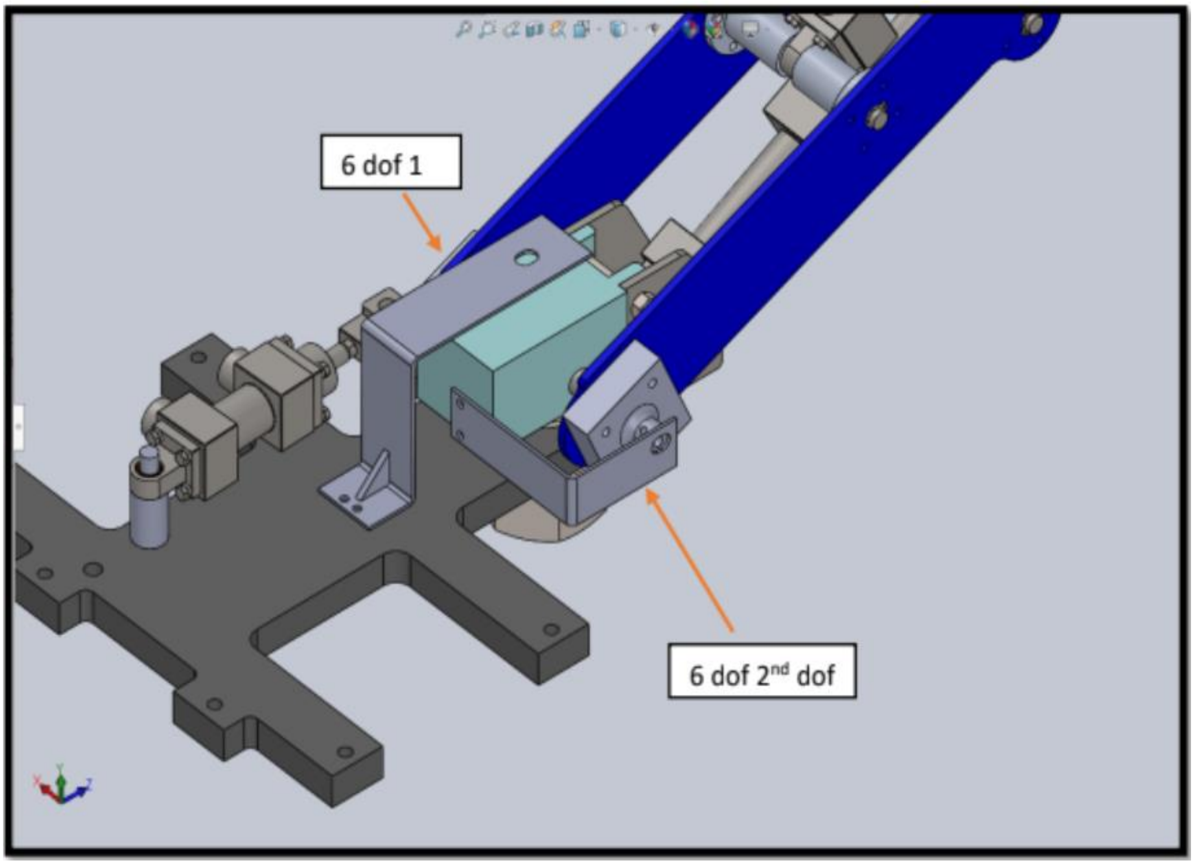
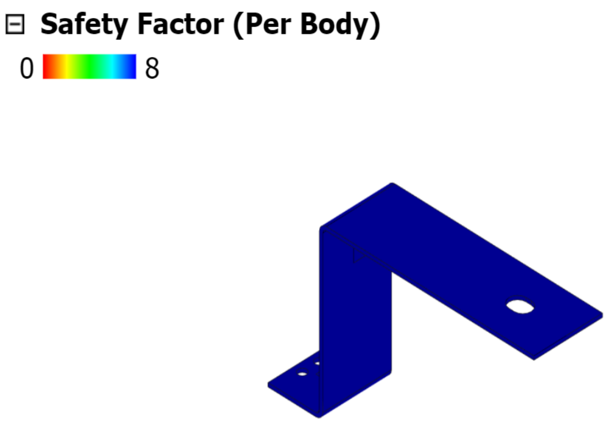
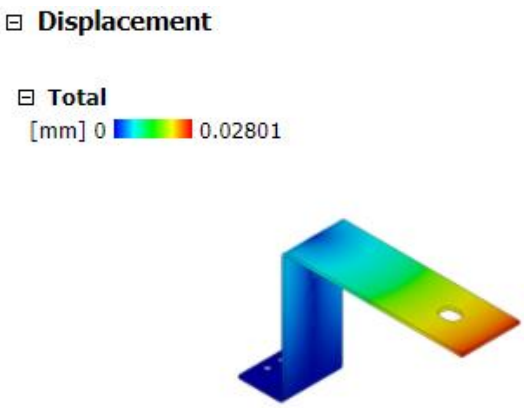


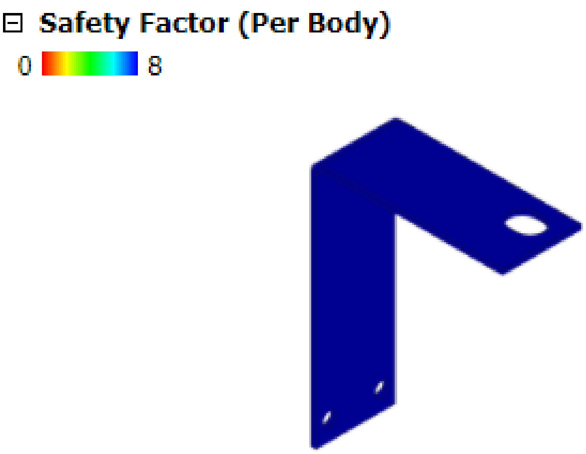
Figure 28: 6DOF Pot Mounting Position CAD

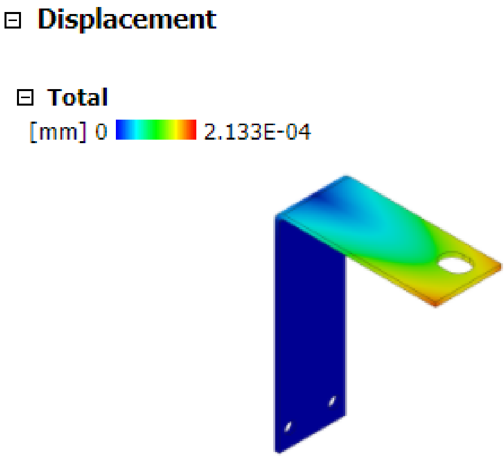
6 DOF Mount 1 Analysis

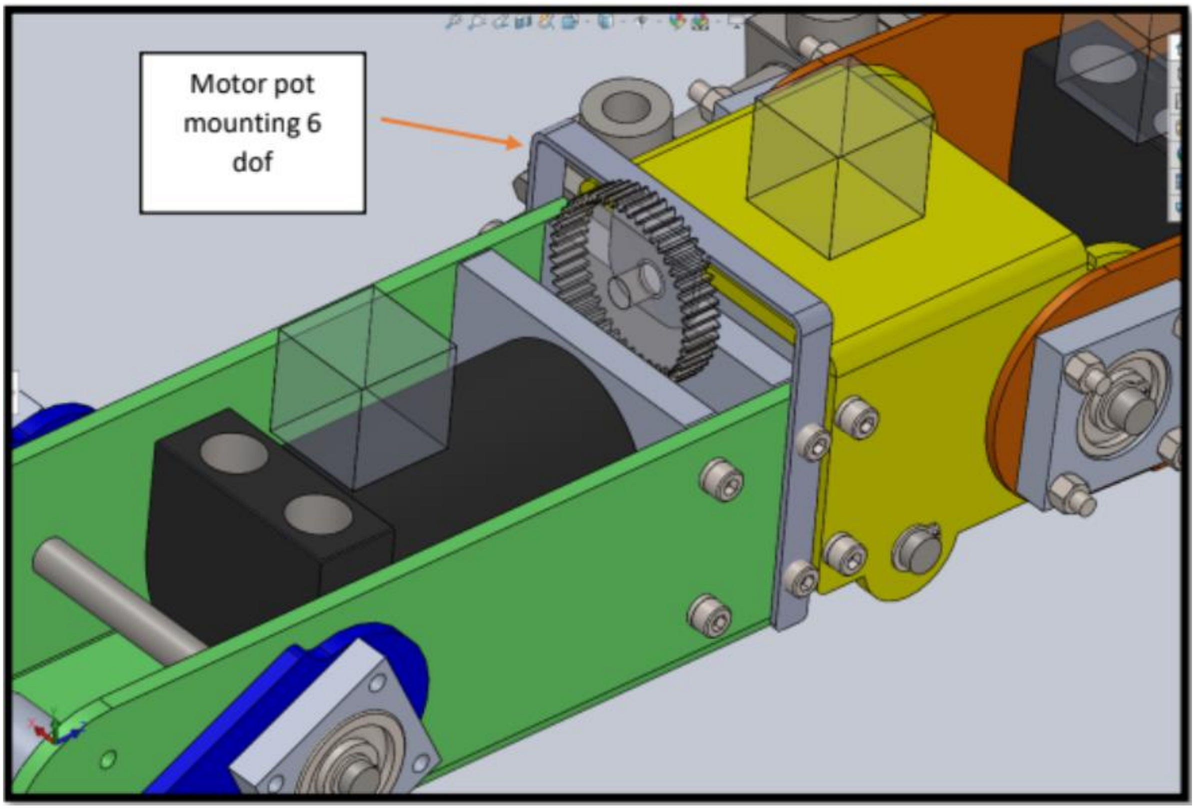




6 DOF Mount 2 Analysis


Safety Factor (Per Body)
0 8


Displacement
Total
[mm] 0 2.133E-04


Motor pot mounting 6 dof

6.1.3. Pinion Gear Design Calculation

Gear Material: Nylon MC 602ST

Bending Stress (β) = 140MPa

Gear Ratio = 2:1

Pressure Angle = 20°

Number of teeth on gear = 60

Number of teeth on pinion = 30

Module = 1.25mm

Radius of gear = R_g = 37.5 mm

Radius of Pinion = R_p = 18.75mm

Lewis Form Factor (Y) = 0.606

Center Distance = $(R_g + R_p)/2$

=56.25 mm

= 15 Nm

Check for Design,

Tangential Force = $(2 * 1000 * T)/ D_p$

= (2 * 1000 * 15)/ 37.5

= 800 N

$V = (\pi * D_p * N)/60$

= 0.196 m/s

$C_v = 6 / (6 + V)$

=0.9683

$C_s = 1.2$

$P_{eff} = (Pt * C_s) / C_v$

= 991.40 N

Considering Bending tooth Failure

Bending Strength (Sb) = m * b * β * y

= 1.25 * 12.5 * 140* 0.606

= 1325.625 N

Checking for Safety,

FOS = Bending Strength (S_b) / P_{eff}

= 1325.625 / 991.40

= 1.604

6.1.4. Gear Pinion Assembly CAD

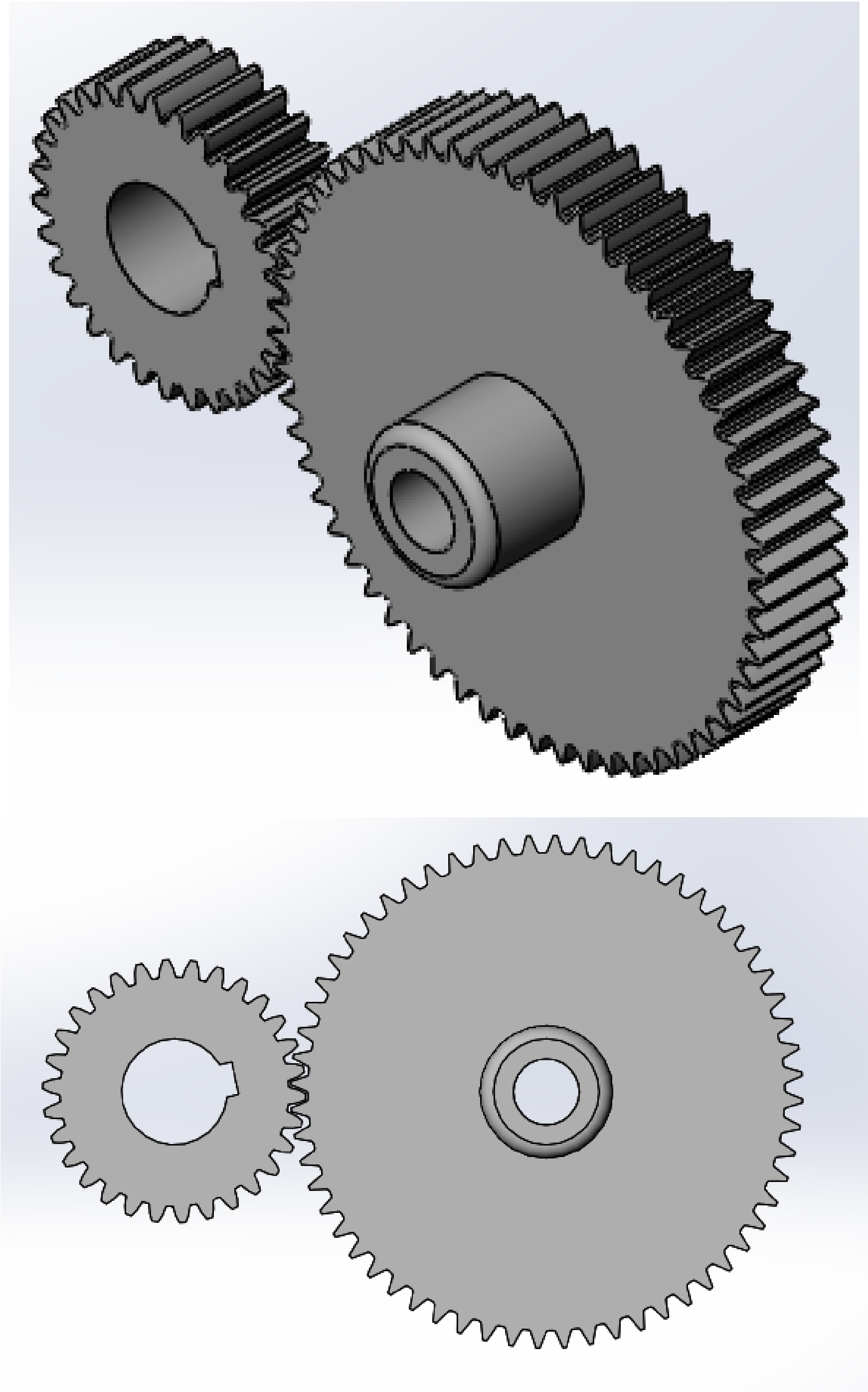


Figure 29: Gear Assembly CAD

6.2ATOS proprietary software for control

The 9 Atos PDCV can be controlled and configured based on the user desired characteristics using the Atos Programming software. To override the factory setting of the pdcv the user has to connect the PDCV to a PC using the M12 5 pin connector provided by atos , Atos's programming software ES-Ws also required. Using this software, we can change all the parameters associated with a PDCV the main parameters in focus are: -

- Scale: Scale function allows to set maximum valve opening at max reference signal value. This regulation allows to reduce the maximum value regulation I front of maximum reference signal.
- Bias and Threshold: This function allows to change the threshold of the reference signal along with dead band.
- Offset: This can be used to set an offset voltage for zero actuation region
- Ramp and Linearization: the ram generator allows to convert a step change of electronic reference signal into smooth time dependent increasing or decreasing of valve opening. Different ramp modes includes-Single ramp input, two ramp input, four ramp input.
- Diether: this function can be used to add filter to input signal for proper flow regulation.

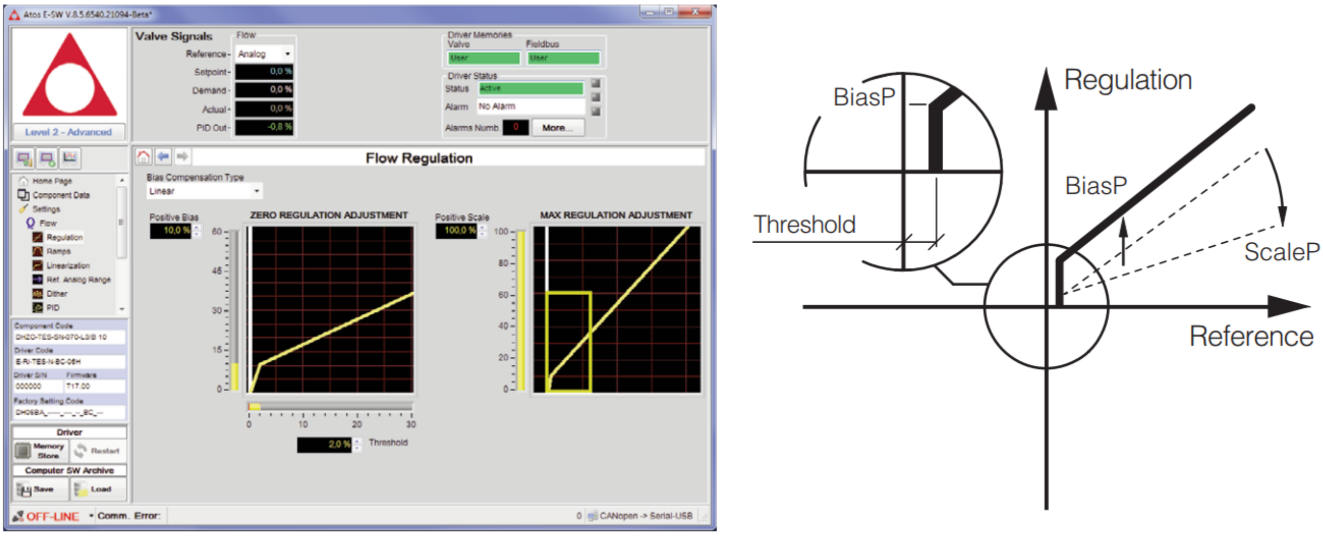


Figure 30: Scale, Bias and Threshold in Atos Software

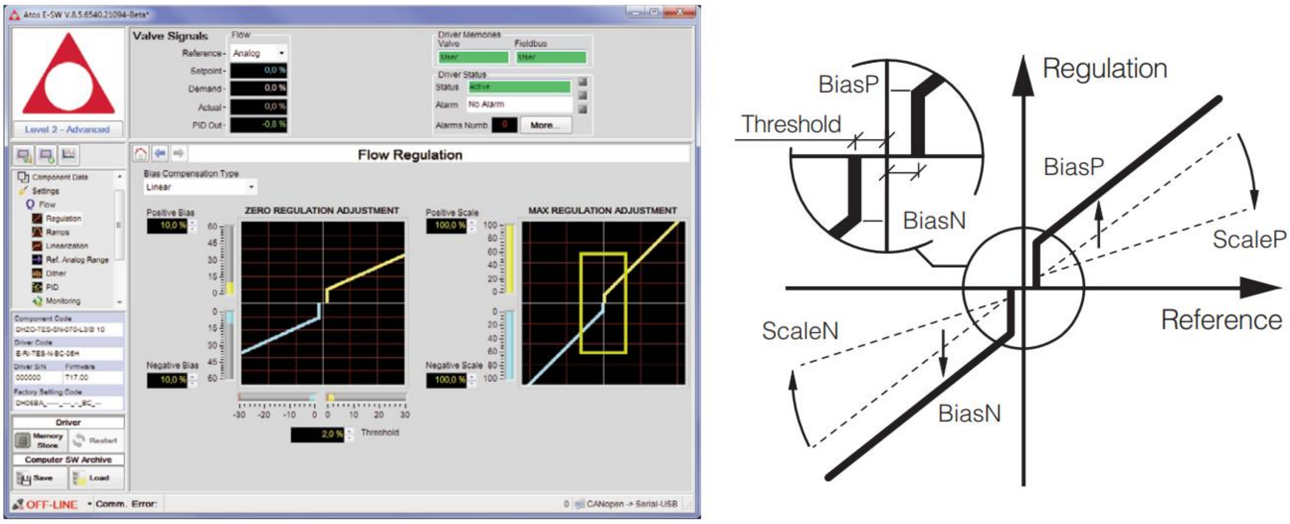


Figure 31: Scale and Threshold in Atos Software

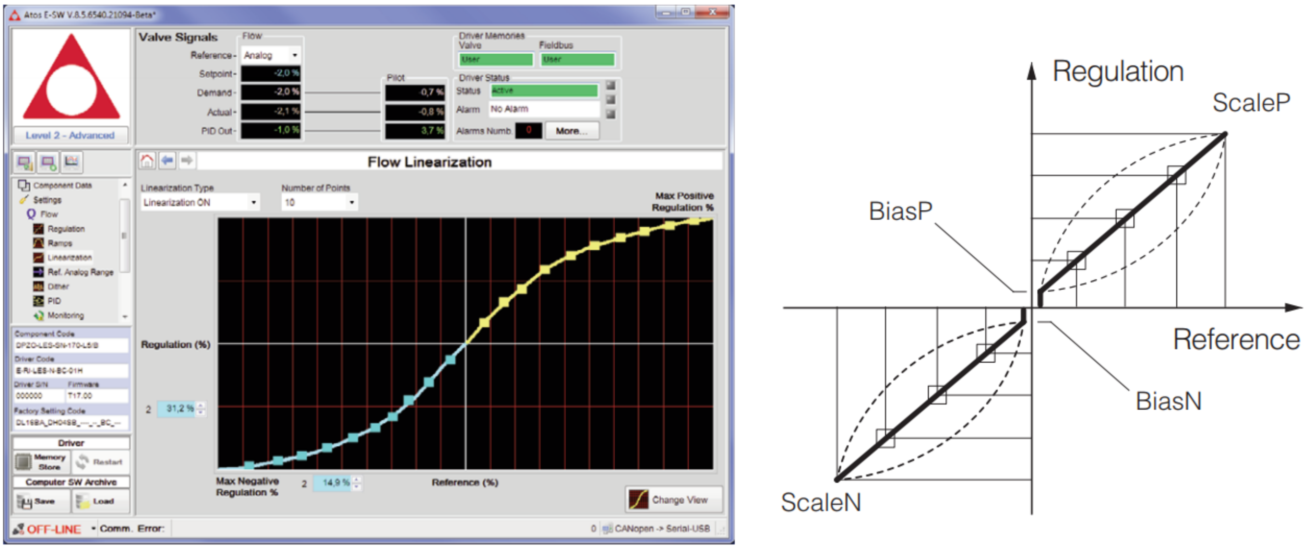


Figure 32: Flow Linearization and Ramp Control in Atos Software

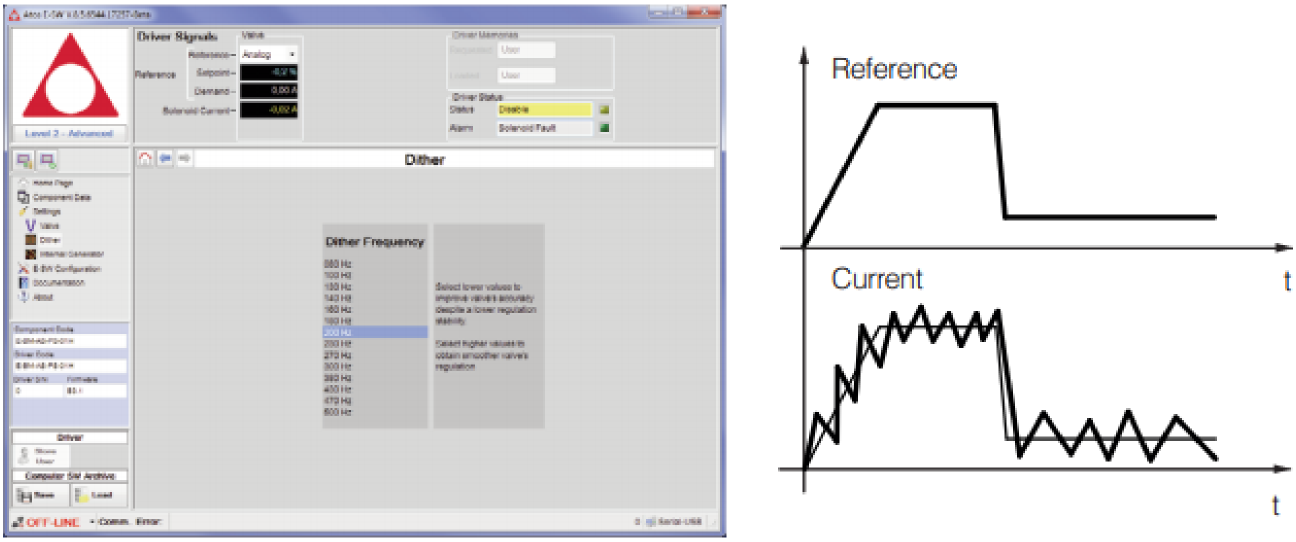


Figure 33: Diether setting in Atos Software

6.3. Umbilical Reel Design

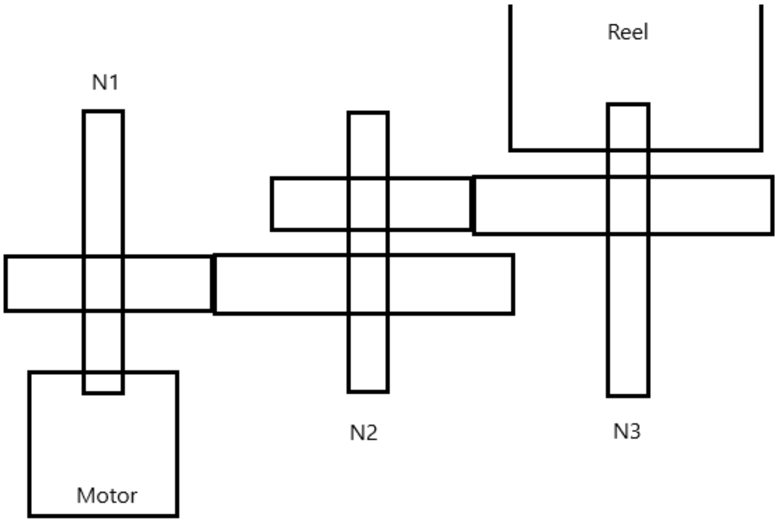


Figure 34: Reel System Concept Diagram

6.3.1Motor Specifications

Figure 35:Selected Motor Specifications

6.3.2 Gearbox Design Calculations

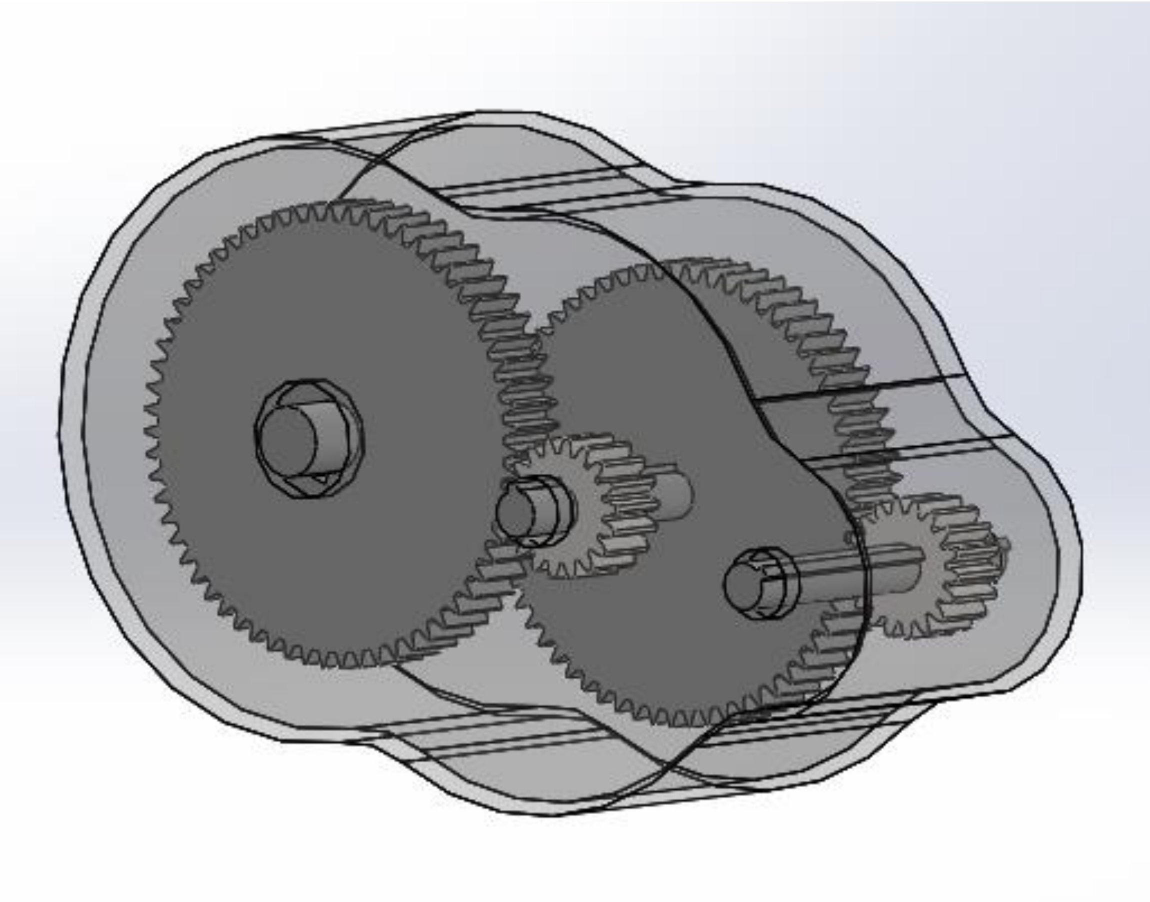


Figure 36: Gear Box Design Diagram

Known Variables:

kW = 0.25

Speed of Motor (N_1) = 120rpm

C_s = 1.5

S_{ut} = 700 N/mm^2 (Plain Carbon Steel 45C8)

FOS= 1.5 (Initially Taken)

BHN = 350

Grade of Machining = 6

Speed of Output Shaft N_3 = 10 rpm

Step 1: Estimation of module based on beam strength

The total transmission is

$$i' = \frac{angular\ velocity\ of\ First\ driving\ gear}{angular\ velocity\ of\ last\ driven\ gear} = \frac{120}{12} = 10$$

$$i = \sqrt{i'} = 3.464$$

For Pressure Angle 20 deg, Minimum Number of Teeth = 18

$Z_p = 18$

$Z_g = i*Z_p = 3.464*18 = 63$

For Ease of Manufacturing, the pinions 1 and 3 are made identical, while gear 2 and 4 are made identical.

$Z_1 = Z_3 = 18$

$Z_2 = Z_4 = 63$

The Speeds of the shafts are as follows,

N_1 = 120rpm

N_2 = 120 *18/63 = 34.285 rpm

N_3 = 34.285 * 18/63 = 9.795 rpm

V = 0.25 m/s (Initially Taken)

$C_v = 3 / (3 + V) = 3 / 3.25$

For N = 18, Lewis form factor Y = 0.308

Since the pair at second stage, i.e., pinion 3 and gear 4 transmit more torque than the pair consisting of gear 1 and 2. Therefore, pinion 3 and gear 4 are to be designed.

$$m = \sqrt[3]{\left[\frac{60 * 1000000}{\pi}\left\{\frac{(kW)Cs(fs)}{z * n * Cv * \frac{b}{m} * \frac{Sut}{3} * Y}\right\}\right]}$$

m = 2.971 = 3mm

Step 2: Gear Dimensions

The first Preference of value of module is 3mm

$d_3' = mZ_3 = 3*18 = 54mm$

$d_4' = mZ_4 = 3*63 = 189mm$

b =10m = 10*3 = 30 mm

Step 3: Factor of Safety using Buckingham's equation

Beam Strength (S_b) = mbσY = 3*30*700*0.308/3 = 6468N

$$Mt = \frac{60*1000000*(kW)}{2*\Pi*N2} = \frac{60*1000000*0.25}{2*\Pi*34.285} = 69631.73 Nmm$$

$$Pt = \frac{2Mt}{d3\prime} = \frac{2*69631.73}{54} = 2578.95 \text{ N}$$

Dynamic Load

For Grade 6,

e = 8 + 0.63Φ

Φ = m + 0.25*(d)$^{1/2}$

e_p = 11.04 µm

e_g = 12.05 µm

e = e_p + e_g = 23.09 µm

Deformation Factor C = 11400 N/mm^2

$$V = \frac{\Pi * d3' * N2}{60 * 1000} = \frac{\Pi * 54 * 34.285}{60 * 1000} = 0.0969\, m/s$$

$$Pd = \frac{21V(Ceb+Pt)}{21V+\sqrt{Ceb+Pt}} = 204.4 \text{ N}$$

P_{eff} = P_d + P_t = 2783.35 N

FOS = S_b/P_{eff} = 2.32

Step 4: Factor of Safety against Pitting

$Q = 2Z_g/Z_g + Z_p = 1.556$

$S_w = bQdp'K = 30*1.556*54*0.16*(340/100)^2$

$S_w = 4644.38N$

FOS = 4644.38/ 2783.35 = 1.66

Step 5:

Gear Dimensions

1. Module = m = 3mm
2. Face width = b = 30mm
3. Addendum = 3mm
4. Dedendum = 3.75mm
5. Fillet radius = 1.2mm

Gears 1 and 3

1. Pitch Circle Diameter = 54mm
2. Addendum Circle Diameter = 60mm
3. Dedendum Circle Diameter = 46.5 mm
4. Number of Teeth = 18

Gears 2 and 4

1. Pitch Circle Diameter = 189mm
2. Addendum Circle Diameter = 195mm
3. Dedendum Circle Diameter = 181.5 mm
4. Number of Teeth = 63

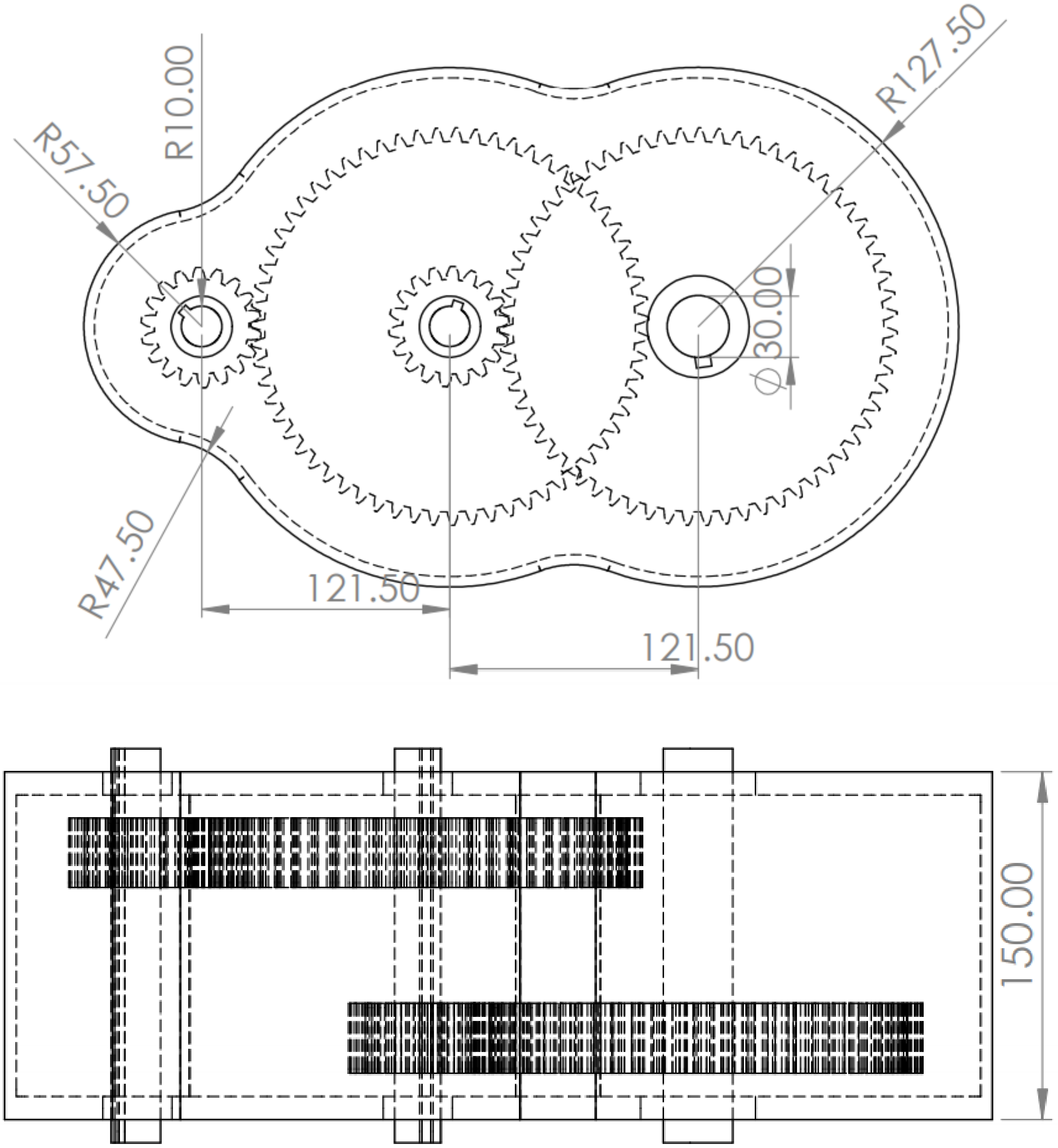


Figure: 37Gear box CAD design

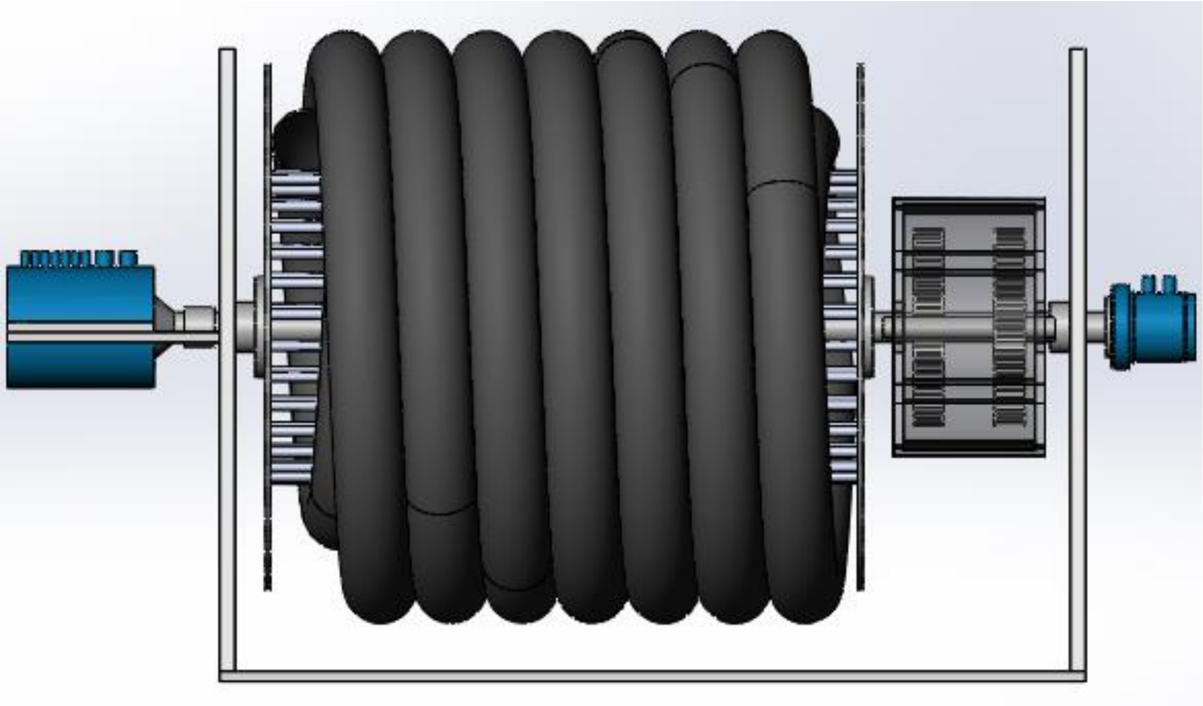


Figure 38: Reel Side View

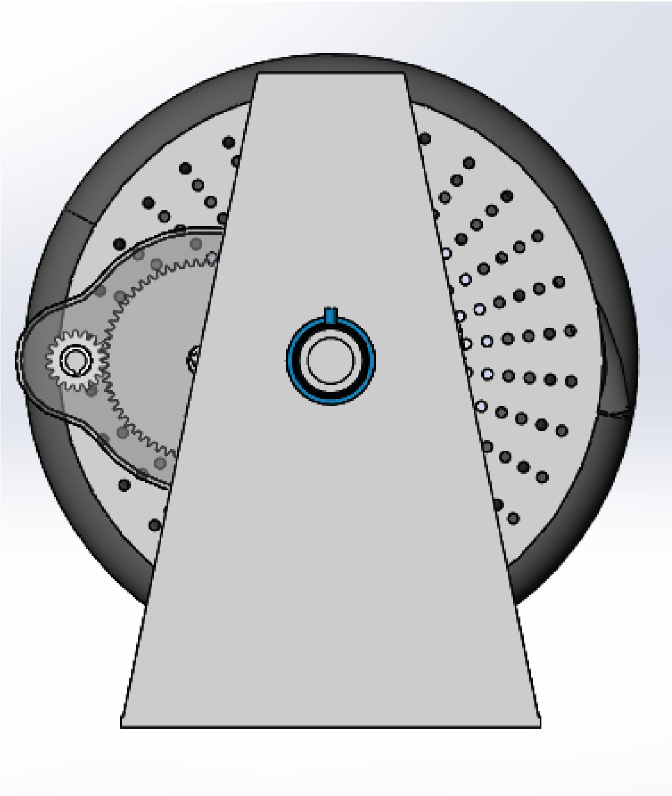


Figure 39: Reel Side View

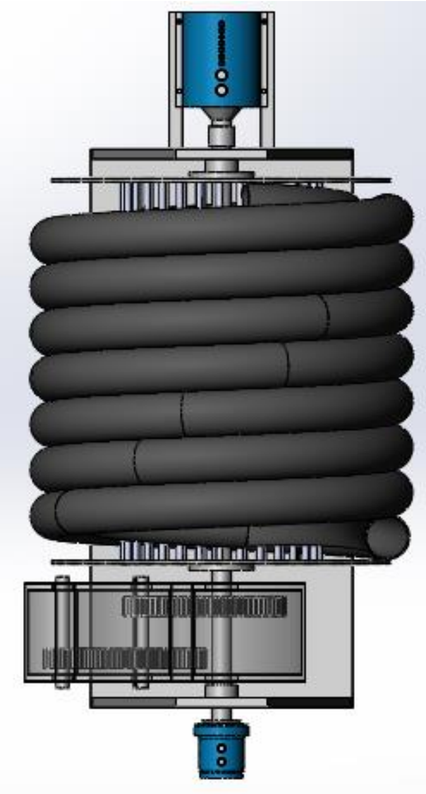


Figure 40: Reel Top View

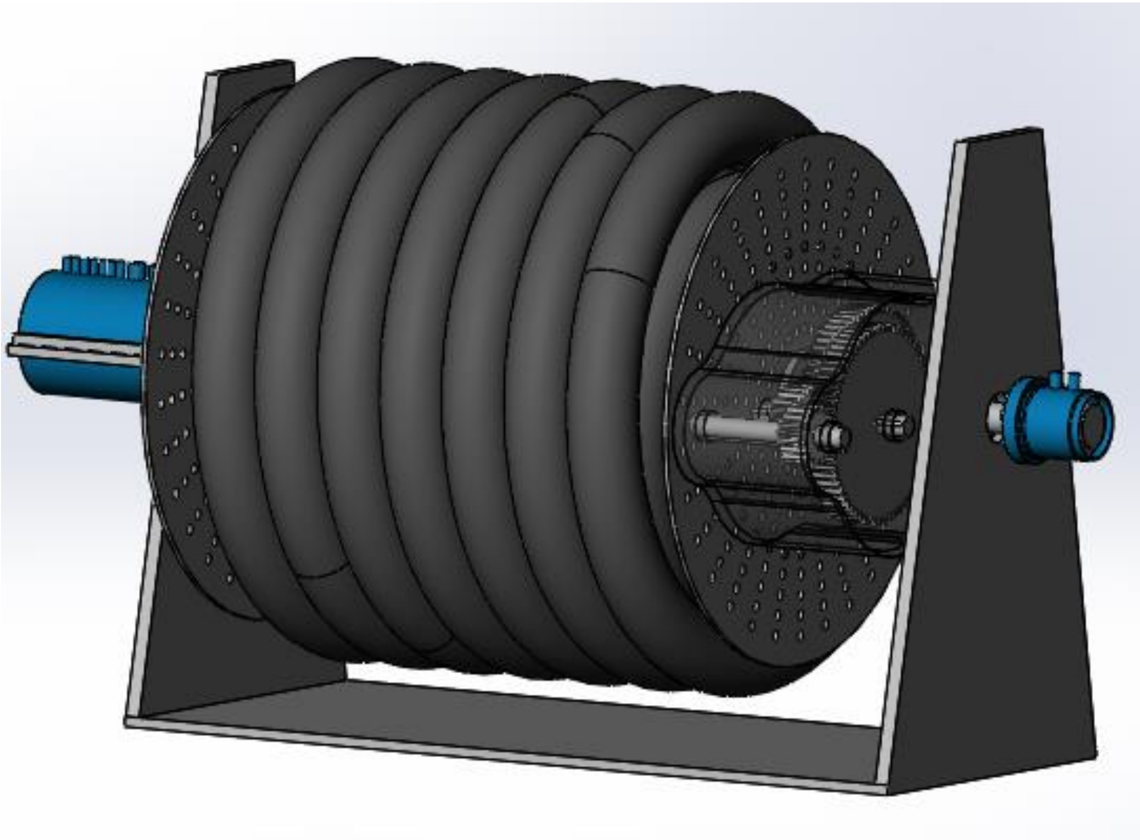


Figure 41: Reel Isometric View

6.4. Hydraulic Circuit Leakage Problem And Solutions

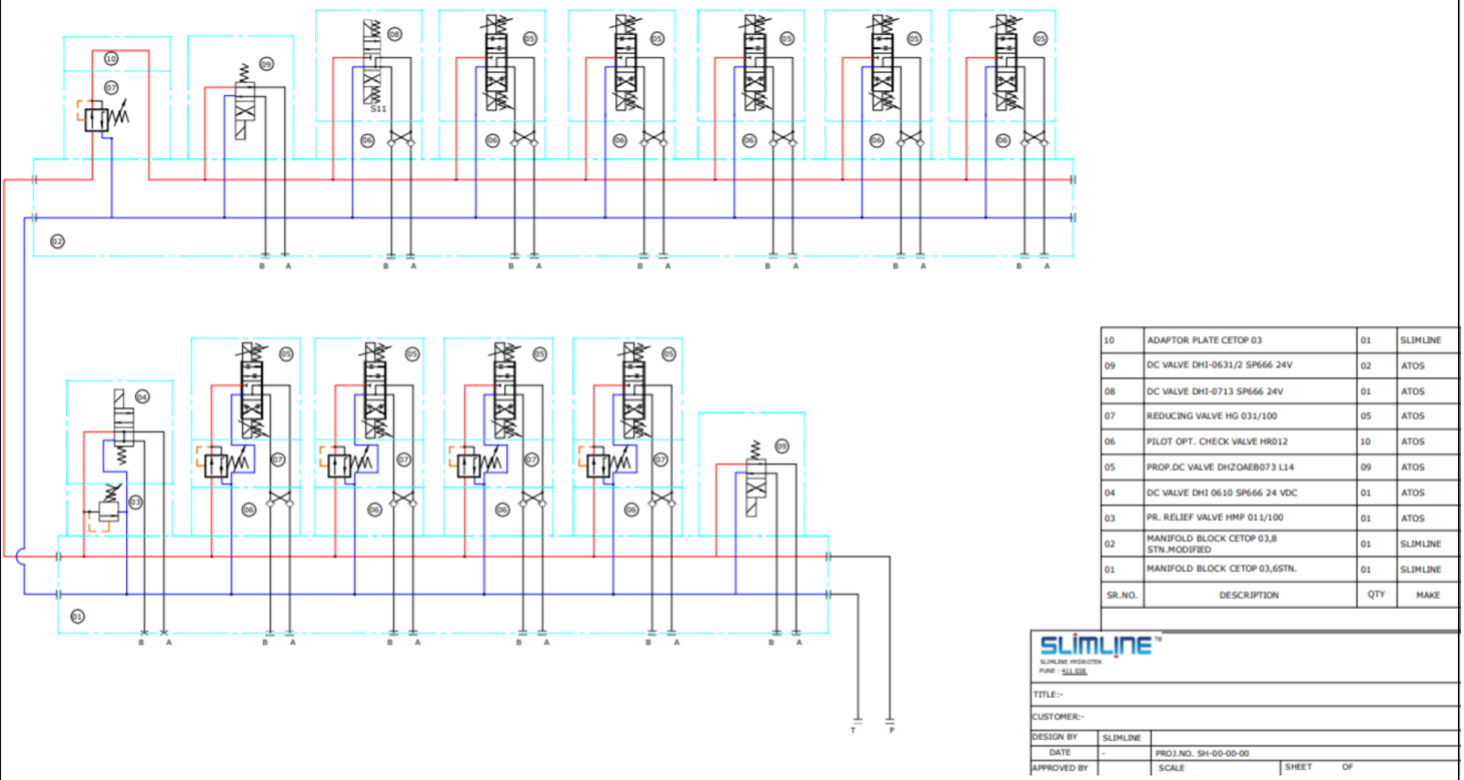


Figure 42: Hydraulic circuit Diagram

6.4.1. Hydraulic Leakage Causes

The main reasons in external leakages in hydraulic circuit are-

1. Low quality fittings.
2. Over tightening of threads.
3. Damage due to normal wear and tear.
4. Excessive stress development due to movement of component.
5. Pressure used beyond rated pressure.
6. Misalignment between fittings and Hoses
7. Excessive bends in flexible hoses

6.4.2. Hydraulic Leakage Solutions

There might be various causes to a external leakage and according by root cause analysis the leakage can be fixed. Some of the solutions tested and implemented are-

1. Changing to a lower diameter hose with same rating to increase flexibility.
2. Use of superior quality fittings.
3. Appling rated torque to tighten a hose.
4. Maintain proper alignment while installing a hose
5. Reducing stress on frequently moving hydraulic components
6. Use of proper sealing agents like o ring or dowty seals

Figure 43: Dowty seal for face fitting Hydraulic joints

BSP Hydraulic Adapter Connections

BSP Size	Tightening Torque	
	Nm	Lbsf/ft
1/8	17	12
1/4	34	25
3/8	47	35
1/2	102	75
5/8	122	90
3/4	149	110
1	203	150
1-1/4	305	225
1-1/2	305	225
2	400	296

Figure 44: Hydraulic fitting Tightening torque ratings

6.4.3. Internal Hydraulic causes and Solutions

Hydraulic leakages can appear in Hydraulic piston-cylinder, Hydraulic Motors and DCV due to

1. Corrosion in internal parts of components
2. Coagulation in oil
3. Presence of air in hydraulic circuit
4. Cavitation in oil due to water presence
5. Degradation of O ring seals

Solutions to fix internal leakages-

1. Check oil sump and change oil regularly
2. Check oil filter and clean if possible.
3. Change O ring if wear can be seen
4. Bleed out hydraulic circuit to remove moisture if present

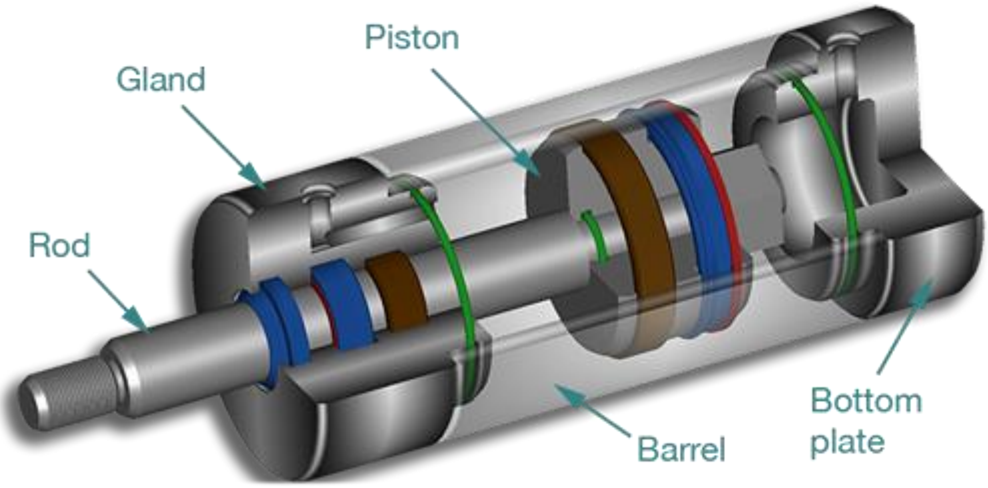


Figure 45: Hydraulic cylinder construction with O ring

7. Conclusion

We have verified and calculated the kinematic models for our specific robot with the help of various tools including the robotics toolbox. We have also verified the physical results further using a feedback system involving potentiometers. As it can be noted from the result tables, the errors between real world physical readings and the simulation results are within acceptable limits and can be used for further control of the robotic arm.

Kinematic modelling of a robot is a key step in designing of a robotic system and in this project, we have covered Forward and Inverse Kinematics. Both methods have different application areas, but these can primarily be used to cross verify the results of each other. This report also covers the geometric modelling approach to calculate the end effector positions. This can be programmed further in MATLAB to make the process more fluid. The PDCV voltage mapping has been completed and a suitable voltage band for operation of the robotic arms has been identified. Finally, PID control of the robotic arms has been tested for two joints and the results have been compiled.

8.References

1. Iqbal J., Islam R., Khan H.,2012. *Modeling and Analysis of a 6 DOF Robotic Arm*. (Canadian Journal on Electrical and Electronics Engineering) 3(6)
2. Deshpande V., George P. M., 2014 *Kinematic Modelling and Analysis Of 5 DOF*

Robotic Arm. (International Journal of Robotics Research and Development (IJRRD))*4(2) 17-24*

3. Denavit J. & R.S. Hartenberg 1955. *A kinematic Notation for Lower- Pair Mechanism Based on Matrices*. (ASME Journal of Applied Mechanics, 215-221.)
4. R K Mittal, J Nagrath. 2005. *"Robotics and Control"*, (Tata McGraw-Hill.)
5. *S. B. Niku. 2001. Introduction to Robotics: Analysis, Systems, Applications.* (Prentice Hall)
6. *Mahalingam S.V 2018, A GUI Based Kinematic Model Development OF 6 DOF Manipulator Using MATLAB.* (International Journal of Engineering Development and Research, 6(4).)
7. Gupta E, Somani S.K. 2019, *Design, Analysis and Kinematics of Robotic Manipulator*: *A Literature Review.* (International Journal of Scientific Research and Review7(5))
8. Cubero S., 2008 *An inverse kinematics method for controlling all types of serial-link robot arms.* (Mechatronics and Machine Vision in Practice, Springer-Verlag, Berlin), *pp. 217-232.*
9. *Robotics toolbox for MATLAB by Peter Corke* (Version9)/MATLAB INC
10. Peter Corke 2011 "*Robotics, vision and control: Fundamental algorithms in MATLAB",* (Springer)
11. Stanislav v., Krejsa J. 2009 *Hydraulic arm modeling via MATLAB Sim hydraulics.* (Engineering MECHANICS*) Vol. 16, No. 4, p. 287–296*
12. Akkar H.R., Amir A. 2016. *Kinematics Analysis and Modeling of 6 Degree of Freedom Robotic Arm from DFROBOT on LabVIEW.* (Research Journal of Applied Sciences, Engineering and Technology)